Generación de Historias de Varios Personajes en Lenguaje Natural con Elementos Narrativos y Diálogo

por Carlos León Aznar

Trabajo de investigación para el

Máster en Investigación en Informática

Especialidad en Sistemas Inteligentes

Universidad Complutense de Madrid

2007

Dirigido por Pablo Gervás Gómez-Navarro

Agradecimientos

Me gustaría agradecer la labor de tutoría que el profesor Dr. Don Pablo Gervás Gómez-Navarro, del Departamento de Ingeniería del Software e Inteligencia Artificial de la UCM (DISIA) ha desempeñado conmigo durante la realización de este trabajo, del cual es director. Asimismo también he de agradecer el trabajo del profesor Dr. Don José María Girón Sierra, del Departamento de Arquitectura de Computadores y Automática de la UCM (DACYA) por ser mi director en el trabajo paralelo a este que se presenta, y que está estrechamente relacionado, de investigación de una plataforma de barcos autónomos cooperantes.

Agradezco también la ayuda constante que me han brindado sendos departamentos (DISIA y DACYA), en especial los grupos de investigación de NIL (Natural Interaction Based on Languaje) e ISA (Ingeniería de Sistemas y Automática), en cuyas plantillas cuentan con investigadores y docentes excepcionales que, en varios momentos de la investigación de este trabajo me han ofrecido, del mismo modo que mis directores, su amable apoyo.

La labor de investigación no pudiera haber sido realizada sin la financiación y ayudas de:

- Beca de Colaboración MEC en el Departamento de Sistemas Informáticos y Programación, Facultad Informática, UCM.
- Indra Sistemas, S. A., a través de una beca en el Departamento de Arquitectura de Computadores y Automática, Facultad de Ciencias Físicas, UCM.
- Proyecto Europeo Automated Digital Fuel System Design and Simulation Process. Departamento de Arquitectura de Computadores y Automática, Facultad de Ciencias Físicas, UCM.

Por supuesto, no puedo olvidar mencionar mi agradecimiento a mi familia y a mis amigos, que en todo momento están a mi lado.

Índice general

Índice de figuras

Capítulo 1

Introducción

Existen, hoy en día, multitud de aplicaciones, sistemas y programas informáticos orientados a la generación de contenidos orientados a ofrecer información a usuarios humanos. La diversidad de este tipo de aplicaciones es muy amplia. Por ejemplo, podemos encontrar sistemas de enseñanza por ordenador (E-Learning), en los que se pretende explicar a un alumno un cierto contenido, como, por ejemplo, la historia de un continente; o aplicaciones dedicadas a modelar ciertos aspectos del comportamiento humano, como simulaciones sociales de grupos de personas, en los que se estudian perspectivas de grupo y de individuo.

En estos sistemas de simulación, lecciones de historia por ordenador, o entornos de realidad virtual inmersiva, por ejemplo, pueden aparecer varias entidades o *personajes* mediante los cuales se transmite la historia. Surge así un modelo de sistema basado en entidades independientes que se relacionan entre sí, con lo que emerge una historia común desde el punto de vista del colectivo de estos *personajes*. En algunos dominios, como estos que se han expuesto como ejemplo, muy posiblemente tenga un gran interés el análisis de esta historia, pudiendo recibir el contenido que se genera en estos sistemas de una forma textual, como una narración o historia, que los humanos podemos entender con gran facilidad.

La Generación de Lenguaje Natural (según sus siglas, GLN), como disciplina científica y parte de la Inteligencia Artificial, se encarga precisamente de esta tarea de traducir datos representados en una máquina según un esquema subyacente no lingüístico en un texto literal legible por humanos [RD00].

En este trabajo de investigación se propone, siguiendo estas ideas, un sistema concreto de Generación de Lenguaje Natural capaz de *crear historias con varios personajes*, en las que se relaten secuencias de *acciones* y *diálogos* entre dichos personajes. Este sistema recibirá, como base de conocimiento, un conjunto determinado de datos que servirán como información a partir de la cual se creará la historia, y que describirán, de manera

exhaustiva, todos los hechos o eventos que se den en una historia. El sistema de generación que se ha creado recibe este conocimiento, y crea, a partir del mismo, y unos objetivos del usuario sobre la información que quiere recibir, una historia en la que los protagonistas sean los personajes y, mediante la narración de sus hechos y la representación de sus diálogos, se transmita a un usuario un conocimiento determinado.

1.1. Motivación y objetivos

La mayoría de los textos que han sido generados automáticamente adolecen de una notable falta de naturalidad. Es sabido que los autores humanos, en el proceso de creación de contenido textual, son capaces de manejar una cantidad ingente de información y de conocimiento relacionados con lo que llamamos sentido común, lo que nos hace capaces de crear contenido más complejo que el que ahora pueden generar las máquinas automáticamente, dada nuestra capacidad de manejar y relacionar la semántica de los hechos que van a narrarse.

Por tanto, cualquier aproximación computacional a la generación de textos que consiga resultados que se acerquen a los que producimos los humanos tiene interés desde varios puntos de vista, ya que hace posible emular, con mayor o menor parecido, tareas humanas que se extienden desde la generación de documentos informativos, con escaso contenido artístico [OM98], hasta tareas más creativas de generación de cuentos, historias o poemas [Ger01].

Actualmente existen varios sistemas capaces de generar textos narrativos a partir de conocimiento. Sin embargo, las aportaciones que podemos encontrar en la literatura son extremadamente específicas y están orientadas a resolver, como veremos y explicaremos más adelante problemas concretos de creación de la estructura del documento, o el formato final. Estos sistemas generalmente ofrecen, para ejemplos concretos, resultados aceptables, pero, en la mayoría de los casos, muy ajustados al dominio, y por lo tanto poco generales.

Por otra parte podemos encontrar sistemas de generación de textos más amplios, que pueden cubrir todo el proceso de generación de un texto, pero están, en su mayor parte, dedicados a fines muy concretos de generación de informes sobre conjuntos de datos. Por ejemplo, de predicción meteorológica, o sobre arquitecturas de programación. Veremos más información sobre estos sistemas en el Capítulo 2.

En este trabajo se pretende tomar un punto de vista diferente en cuanto a la generación de textos se refiere. Por un lado, la intención es crear textos capaces de transmitir una historia específica usando personajes que interactúan entre ellos, dialogando o llevando a cabo tareas comunes, en la que puede estar, explícito o implícito, cierto mensaje. La historia, por

tanto, es el núcleo, no sólo la información. A pesar de que, evidentemente, transmitir una historia es transmitir información, y, por tanto, la generación de historias se engloba dentro de la generación de lenguaje natural, existen ciertas características que hacen diferente la creación de textos informativos con respecto a la creación de textos narrativos. De nuevo, en el Capítulo 2 examinaremos esto con más detalle, atendiendo al estado del arte en estas disciplinas.

Resumiendo, lo que promueve esta investigación es aportar un sistema general de creación de historias narrativas en las que puedan aparecer varios personajes, y que tenga la capacidad de mostrar no sólo la actividad de estos personajes por separado, sino también la relación que entre ellos pueda surgir. Para esto, se aplican partes de narración textual y diálogos, tal y como se comenta en el Capítulo 3.

Es importante notar que no nos preocupamos, en general, de cuál sea el mensaje interno de la historia, si lo tiene. La labor de este proyecto de investigación no es el estudio de la semántica de la historia como tal, sino de crear una historia a partir de unos datos de una forma coherente. Por supuesto, el significado de estos datos influye determinantemente en el esqueleto de la historia que creamos, pero no es asunto de este trabajo evaluar el contenido último, su corrección o su interés, ya que el objetivo consiste en generar una historia que tenga unas propiedades de legibilidad y de transmisión de contenido lo más cercanas posibles a las que tienen los textos generados por humanos.

1.2. Problemática de la Generación de Lenguaje Natural

Existen diversos métodos a la hora de hacer que una aplicación genere mensajes en lenguaje natural, con distintos grados de complejidad y flexibilidad. En un extremo del espectro, tenemos los tradicionales "textos enlatados" (*canned texts*): ante determinadas situaciones, el sistema muestra mensajes a partir de plantillas predefinidas directamente en el código del programa. Esta solución es la menos flexible y más dependiente del dominio de aplicación, pero proporciona resultados bastante aceptables y, en ocasiones, suficientemente correctos para sistemas sencillos. En el otro extremo, tenemos la generación de lenguaje natural basada en conocimiento (*deep generation* o generación profunda): el sistema genera todos los mensajes de forma dinámica, basándose en el conocimiento lingüístico y el conocimiento del mundo en forma de ontología o jerarquía conceptual de que dispone. Nuestro sistema se halla dentro de este grupo. El comportamiento del sistema es mucho más flexible, adaptable y ampliable, si bien su desarrollo es, en la mayoría de los casos, mucho más complejo. Algunos sistemas actuales utilizan soluciones híbridas, enlazando texto enlatado o plantillas (fragmentos

de texto que se utilizan repetidamente en distintos documentos del mismo dominio en los que varían sólo algunos de sus elementos, por lo que pueden ser fácilmente parametrizables para adaptarlos a distintas situaciones) con fragmentos generados de forma dinámica a partir de una ontología propia del dominio.

El problema fundamental de los generadores actuales es que su construcción es compleja y está muy ligada al dominio en el que se aplican, con lo que la posibilidad de reutilización de estos programas de ordenador es escasa o prácticamente nula. Desde el punto de vista arquitectural existen numerosas formas de organizar un sistema de Generación de Lenguaje Natural [DHZ95, Rei94]. En la literatura podemos encontrar diversas arquitecturas con grandes diferencias respecto a la división en módulos y la topología de conexión entre ellos, y cada una de ellas presenta sus correspondientes ventajas e inconvenientes. Tener que moverse en un espectro tan amplio de posibilidades de diseño como el descrito hace que esta característica del campo sea especialmente grave.

La conclusión es que cualquier desarrollador de una aplicación de generación deberá considerar un amplio rango de soluciones arquitectónicas, ya que cualquiera de ellas podrá ser relevante para aspectos particulares de su problema. Sin embargo, el desarrollador que se plantea añadir un módulo de generación de lenguaje natural a una aplicación existente no contempla la posibilidad de dedicar largo tiempo al desarrollo a medida de su solución. Para conseguir que el uso de este tipo de aplicaciones se extienda, se hace necesario el proporcionar algún tipo de esqueleto que facilite esa labor, posiblemente que incluya distintas opciones ya preparadas de módulos capaces de resolver tareas concretas, utilizando distintas técnicas, y que resulten fácilmente enlazables para obtener una solución operativa.

1.3. La importancia de la información

La solución que se propone en este trabajo consiste en un conjunto de ideas y algoritmos organizados en una arquitectura de programación con los que se pretende aportar nuevas ideas y soluciones a un problema antiguo (desde el punto de vista de la corta vida de la Informática), que es la Generación de Lenguaje Natural. Como se irá viendo a lo largo del desarrollo de esta propuesta, se ha invertido gran esfuerzo en la investigación de las raíces de los problemas que existen hoy en día en la tarea de generación de textos. Estas raíces no son tanto de carácter algorítmico como de adquisición, organización y uso de la información.

Sin ser el objeto de este estudio, merece la pena dedicar unas líneas al concepto de *información* o *conocimiento* que usamos en este trabajo. Podemos decir que los datos se convierten en conocimiento cuando los sistemas encargados de procesar y usar estos datos son capaces de utilizarlos con una

semántica dada. De este modo, un valor del estilo de "$tiempo = 12:30$" es simplemente lo que denominamos como *dato* hasta que con él se alimenta a un sistema capaz de reconocer, mediante el algoritmo que sea, que algo ha ocurrido a las doce y media, por ejemplo.

Una muy buena parte de las alternativas existentes para solucionar los problemas de la generación de lenguaje natural orienta su esfuerzo precisamente a abordar los problemas que conlleva la falta de información disponible para generar un texto respecto de la que es capaz de usar el ser humano. Este trabajo se plantea de la misma manera: dados los escasos recursos de conocimiento que disponemos respecto a los que usamos como humanos, sus capacidades lingüísticas intentan ser emuladas mediante un uso diferente de la información, y mediante la adición de nuevas estructuras que contienen información para el sistema. Con esto se intenta que las carencias que aparecen en los ordenadores sean suplidas.

1.3.1. La información en la Generación de Lenguaje Natural

La Generación de Lenguaje Natural tiene como propósito la transmisión de una información determinada almacenada en un ordenador hacia un destinatario humano. La comunicación humana, entendiendo como tal la transmisión de información entre personas, es un campo que ha evolucionado paralelo al hombre, y, de hecho, ha sido uno de los motores del avance social humano.

Sin embargo, transmitir esta información no es una tarea trivial. Los humanos no nos ceñimos únicamente a enunciar los datos concretos que queremos que conozcan los receptores de la comunicación, sino que "moldeamos" esta información, preparando los mensajes de modo que, junto con el contexto y otras formas de comunicación, cuando se dan (comunicación no verbal, maquetado de un texto), se crean unidades de transmisión llenas de significado. Además, el discurso de los mensajes complejos añade aún más semántica, y en este tipo de mensajes creamos una gran cantidad de "submensajes" que ayudan a entender el contenido principal que quiere transmitirse (por ejemplo, la introducción en un artículo científico respecto del núcleo de la aportación de dicho artículo).

El objetivo de este trabajo es proponer soluciones a estos problemas recién expuestos. En general, podemos decir que la generación de lenguaje natural en computadores es la emulación de la comunicación ser humano. Por tanto, vamos a aplicar técnicas de tratamiento de información, como se explica a lo largo de toda esta memoria del trabajo, para conseguir resolver los problemas que surgen en la GLN.

1.3.2. ¿Qué necesitamos para generar una historia?

Consideramos que generar una historia a partir de un conjunto de datos es el proceso de crear un contenido textual que transmita una información presente en dicho conjunto. Sin embargo, una historia contiene más información que la presente en los datos. La información presente en una historia que no está en el conjunto de datos puede aparecer de forma implícita, haya sido incluida en la historia por un humano o de forma automática por un máquina. Podemos dividir esta información en cuatro grupos o tipos:

- *Información de filtro del contenido.* Esta información determina qué hechos deben ser contados, y cuáles no. Las operaciones que se aplican están, como normal general, incluidas dentro del conjunto de operaciones de DETERMINACIÓN DEL CONTENIDO. Si, en una historia, apuntamos todos los datos de los que disponemos, crearemos una obra en la que hasta el más mínimo detalle quede explícito, creando un conjunto de datos textuales que dista muy poco de un archivo de datos, con lo que la legibilidad de los textos se reduce enormemente, y, además, deja de aprovechar el conocimiento contextual que los humanos tenemos.

- *Información del orden de los acontecimientos.* Es la información que se obtiene aplicando las operaciones de *planificación del discurso.* Esta información ordena los hechos, dando una secuenciación que expresa el contenido del texto de una manera coherente y más inteligible. Sin este tipo de operaciones los textos y las historias generadas se convertirían en meros conjuntos deslavazados de datos que, a pesar de que probablemente tuvieran relación entre sí, al carecer de un orden de transmisión del escritor de la historia hacia el lector de la misma, pierden su funcionalidad como mensaje de comunicación.

- *Información de la relación entre los hechos.* Cada hecho, aislado, tiene significado por sí sólo. No obstante, una narración necesita coherencia en el sentido de que es necesaria la relación que cada hecho tiene con los demás. Una narración tiene algunas relaciones explícitas entre hechos, y una gran multitud de relaciones implícitas. A lo largo del desarrollo se verá la importancia que tiene la relación que el discurso halla o expone de los hechos, y cómo estas influyen en la percepción del lector y oyente durante el transcurso de la historia.

- *Información de la agregación de los hechos.* Las operaciones de *agregación*son las encargadas de generar esta información. La información de la agregación junta unidades semánticas (nombradas en este trabajo como *hechos* o *átomos*) en unidades que podemos llamar *frases*, y contienen que añaden una estructura a la información que hace la historia más cercana a los mecanismos de comprensión humanos. En todos los

lenguajes naturales humanos se ejercen operaciones inconscientes, pero claras, de agregación. Sin ella, no tendríamos la subordinación o la coordinación que tan fácil hacen para nosotros la transmisión hablada de la información.

Por lo tanto, necesitamos esta información para escribir, con un computador, una historia coherente. Es posible dar esta información explícitamente, de muchas maneras. Aplicando reglas de plantilla, usando gramáticas de producción, etcétera. En el Capítulo 2, en la que hacemos un estudio sobre las tecnologías sobre las que hemos apoyado y hecho avanzar la presente investigación, se da más detalle sobre esto.

1.3.3. ¿Qué debemos narrar en la historia?

Según las ideas presentes en [RD00], un sistema de generación de texto en lenguaje natural tiene como finalidad representar la información estructurada de las máquinas de una forma que un humano pueda comprenderla con facilidad. Como se ha comentado anteriormente, este trabajo está orientado a la generación de historias coherentes en lenguaje natural a partir de una gran cantidad de información de sistemas en los que actúan varios personajes. La disponibilidad de esta información nos permite crear una cantidad potencialmente infinita de historias a partir de ella, ya que es posible generar de manera lineal todo el contenido, narrar solamente la vida de un personaje, contar las relaciones de una familia, y demás. Cualquier contenido que se cree puede ser válido desde el punto de vista semántico y de la coherencia.

Por tanto, el primer asunto que debemos tratar en la generación de historias es saber qué es lo que queremos contar, o, dicho de otro modo, cuál es la información que deseamos transmitir a otra persona. Este es un problema que no podemos resolver a priori. En un conjunto de datos, sea del tipo que sea, no sabemos cuáles son los que le van a interesar al lector. En los propios datos no podemos encontrar esta información, por ser ésta de índole subjetiva, entendiendo como tal el hecho de que el sujeto puede tener diferentes necesidades, intereses o curiosidades, incluso sobre el mismo conjunto de datos en diferentes momentos. Un humano, habiendo recibido un conjunto de datos que describan de una manera estructurada el comportamiento durante un intervalo de tiempo de un colectivo de personajes será incapaz de crear un texto coherente si no dispone de un objetivo sobre la narración. Este objetivo puede estar dado por él, habiendo, por ejemplo, decidido que va a elegir un protagonista, y va a hilar el discurso sobre su vida; o determinado desde fuera, por ejemplo si se le ha solicitado que cuente la historia de todos los personajes que han llevado a cabo un proyecto común.

Del mismo modo operan las máquinas, como emuladoras del hombre. Es necesario disponer de un mecanismo que decida qué historia contar a partir de los hechos que se encuentran disponibles en la entrada del sistema. Si a

un humano se le dan una gran cantidad de datos y se le pide que cuente una historia a partir de ellos, es muy probable que necesite saber "qué parte ha de ser contada". Como una simple clasificación de las posibilidades que tiene un sistema informático de orientar la generación de historias desde el punto de vista que acabamos de explicar, proponemos la siguiente.

- **Imposición externa**: Entendemos por tal aquella en la que un agente externo (el hombre u otro sistema informático, por ejemplo) establezca de antemano, como entrada del sistema de generación de lenguaje natural, qué ha de ser contado. Así, el sistema únicamente estaría encargado de ejecutar un algoritmo que cumpliese que la salida textual siga los requisitos de la entrada especificada externamente.

- **Decisión orientada por contenido**: En este grupo englobamos todos aquellos mecanismos informáticos en los que el ordenador decide, a partir de los datos de entrada únicamente, qué debe ser narrado. Cualquier algoritmo que sea capaz de dar esta información puede ser válido, desde un sistema basado en reglas hasta un complejo sistema de búsqueda por estados.

- **Decisión creativa**: En vez de disponer de reglas deterministas que permitan decidir qué ha de ser narrado, el sistema podría usar técnicas de aleatoriedad para establecer el hilo conductor de la historia, de modo que la historia resulte creativa.

Esta taxonomía que exponemos, por lo tanto, nos muestra tres grupos posibles de técnicas para la determinación del hilo de la historia. Como es evidente, estos grupos no son exclusivos, y es posible crear un algoritmo que presente características de cualquiera de los grupos de la clasificación. Para realizar esta división hemos seguido la idea de separación hombre máquina en tres niveles. Es decir, hay un nivel en el que el hombre decide el absoluto sobre lo que ha de generar el ordenador (la *imposición externa*), otro en el que la decisión está tomada a medias entre el hombre y la máquina (*decisión orientada por contenido*), y un nivel final (*decisión creativa*) en el que sólo la máquina establece lo que será narrado. Desde luego, las máquinas no "deciden" en el sentido humano de la palabra, y son los algoritmos en ellas implementados los que llevan a comportamientos que llamamos de "decisión". A continuación exponemos con más detalle cómo se han afrontado estos posibles tipos en el proyecto que se presenta.

Por supuesto, hay que notar que es perfectamente posible alternativas que contemplen varias de estas opciones al mismo tiempo, permitiendo que el usuario decida sólo una parte de lo que contar, o usar reglas para guiar la producción creativa, etcétera.

Imposición externa

Es posible orientar la narración especificando qué tipo de historia se requiere. Si a un humano se le ofrece una cantidad de datos grande, es pertinente pedirle que cuente "la historia del héroe" o "la historia más alegre". A la hora de crear un sistema automático de generación, es muy útil conseguir que el programa también tenga esta funcionalidad.

¿Cómo conseguirla? El primer paso es dotar al sistema de una herramienta que permita especificar tipos de historias, de modo que un humano pueda establecer la información que quiere recibir y transferir sus objetivos a la máquina. Así será capaz el usuario del programa de guiar el proceso de generación de historias.

Sin embargo, debemos llegar a un compromiso entre control de la generación y trabajo necesario del hombre. Si fuese necesario especificar completamente todos los puntos de la historia, no habría utilidad en el sistema de generación de textos. Por otro lado, si las capacidades de especificación no son suficientemente potentes, no permitimos que el usuario reciba del ordenador el tipo de historia que desea recibir.

Decisión orientada por contenido

Llamamos *mundo* al conocimiento que el sistema generador de narraciones tiene de un dominio dado. Este mundo contiene datos que representan la información con la que puede trabajar el sistema. Cuando un usuario conoce el mundo, puede decidir qué parte de ese mundo desea obtener como narración, como hemos explicado en el apartado anterior.

De la misma manera, es posible dotar a un sistema informático de un modelo de decisión de historias que no requiera, por tanto, de la labor humana para establecer qué información ha de ser transmitida. Este modelo, como se ha comentado antes, puede ser todo lo simple o complejo que se necesite. Por ejemplo, un sistema de reglas que, a partir de la ciertos parámetros que cumpla la información contenida en el conocimiento del mundo como condiciones de cada regla elija una información objetivo puede servir.

No obstante, esta aproximación no siempre es válida, aunque el sistema sea de mucha calidad. El usuario puede no querer dejar que el sistema decida cuál es la información más relevante, y es posible que necesite información que no se considera importante. Por ejemplo, si se usa un sistema narrador para resumir las operaciones de un usuario en el proceso de depuración de un programa, y el usuario quiere indagar en los aspectos de más detalle de su operación o la del depurador, una aplicación que filtre los hechos menos influyentes puede no ser útil.

Decisión creativa

En la Sección 2.3 damos referencias e información sobre la posibilidad del comportamiento creativo en las máquinas. El hecho de que pudieran llegar a comportarse, durante la producción de textos de forma automática, de la misma forma que los hombres, está fuera del alcance del estudio de este trabajo de investigación. No obstante, a pesar de que, como comentamos más adelante, la creatividad no tiene una definición exacta ni formal, sí es cierto que existen ciertos patrones que los humanos tildamos de "creativos". Por tanto, si hacemos que las máquinas generen *artefactos* que, si hubieran sido desarrollados por humanos, serían llamados "creativos", podemos concluir que hemos alcanzado la Creatividad Computacional.

Por tanto, y dado que este objetivo no sólo es posible, sino que ya existen varias aportaciones sobre ello [PG06], tenemos la opción de delegar la decisión sobre qué ha de ser narrado en los ordenadores, que, usando uno u otro algoritmo, e información no directamente relacionada con el objetivo de la historia, pero sí con el contexto en el que va a ser relatada, habrían de decidir qué historia van a contar.

1.4. Viabilidad

Idear, desarrollar, implementar y desplegar un sistema de lenguaje natural, en general, es una tarea complicada. Existe una vasta cantidad de problemas e inconvenientes, varios relacionados con los sistemas de programación en general, y algunos otros que simplemente aparecen en el campo de la generación de lenguaje natural, que han de ser atendidos. Estos problemas muchas veces disuaden al usuario final de la utilización de sistemas de Inteligencia Artificial tan complejos, y son asuntos que no han de ser descuidados. A continuación enumeramos algunos de ellos, los que consideramos más influyentes y dignos de atención:

1.4.1. Capacidad del sistema

Evidentemente, el primer aspecto a considerar es la capacidad del sistema para producir textos. A pesar de que la GLN es un área de investigación en la que ya se han hecho sustanciosos progresos, no hay que olvidar que es ciencia en período de desarrollo, y las posibilidades de estos sistemas no son absolutas. No podemos, hoy en día, crear un programa de GLN que escriba novelas complicadas, llenas de figuras retóricas, de gran belleza y sentido.

Sin embargo, sí que es posible (y, de hecho, se hace) usar sistemas de generación automática para elaborar documentos e informes de áreas específicas, que sirven de apoyo para la labor humana.

1.4.2. Coste

Elaborar un sistema que crea textos de manera automática es una tarea muy compleja, en términos de recursos humanos y técnicos. Por esto, es imperativo realizar un estudio previo de costes en el que quede evaluado cuál va a ser la ganancia que piensa obtenerse con la automatización de la generación de textos por máquinas, teniendo en cuenta cuántos recursos se planea emplear en el desarrollo, despliegue y pruebas.

Para explicar con un ejemplo, si una máquina capaz de producir informes en una determinada área de trabajo piensa ser usada para crear mil páginas al año, y cada página cuesta, considerando de forma global el sistema, cien euros, tenemos que cada página nos ha costado, calculando, cien mil euros para mil páginas. Si un operario humano emplea veinte euros en escribir una página, tendríamos un gasto de veinte mil euros al año. Sin embargo, habría que considerar si el humano es capaz de generar mil páginas al año.

En resumen, preparar para producción un sistema de generación de lenguaje natural impone la elaboración de un estudio de coste, ya que un programa de tales características es tan caro desde varios puntos de vista, que supone un riesgo grande el empleo de uno de ellos.

1.4.3. Aceptación de las máquinas

A pesar de que los ordenadores se han convertido en una herramienta de trabajo prácticamente necesaria para una cantidad enorme de diferentes disciplinas profesionales, los humanos aún tendemos a recelar de las posibilidades de los ordenadores. Además, mucha gente se ha acostumbrado a usar las máquinas de un modo determinado, y rehúsan aprender nuevas características y posibilidades de operación de los sistemas más modernos. Por esto, el avance de las tecnologías informáticas en general se ve ligeramente retrasado.

Por otro lado, muy a menudo los humanos suelen presentar una alta reticencia a aceptar responsabilidades de los productos o servicios que ellos no han realizado. Este aspecto se pone más de manifiesto cuando se trata de ordenadores. Por ejemplo, un doctor difícilmente va a firmar un documento sobre consejos dietéticos que no ha redactado de su puño y letra.

1.4.4. Aplicación a nuestra investigación

Para la investigación que nos ocupa, de estos tres puntos aquí analizados sólo hemos de preocuparnos, realmente, por el primero. El objetivo final del trabajo es conseguir que las capacidades de generación de los sistemas de GLN sean mayores. Por otro lado, para estudiar la viabilidad de nuestro trabajo en particular, saber cuáles son las capacidades actuales de estos programas, para fundar nuestra investigación, es más que fundamental. A priori podemos decir que nuestro trabajo es viable, puesto que se basa en

técnicas conocidas, y los resultados han puesto de manifiesto que es posible utilizarlo en varias áreas, por el momento, de investigación.

Sobre el coste, no hemos de tener cuidado especial en este trabajo (más allá de lo que sea posible implementar y estudiar, claro está), pero podemos olvidarnos de la comparación de la elaboración manual de textos, o del despliegue para producción, ya que son aspectos que están fuera de este estudio.

La aceptación que el usuario muestre de las historias con diálogo que sea capaz de generar nuestro sistema nos servirá, como comprobaremos a lo largo de la descripción del trabajo, de sistema de evaluación de la calidad de las historias generadas.

1.5. Estructura de la memoria del trabajo de investigación

La presente memoria sobre el trabajo de investigación que se ha realizado está dividida en cinco capítulos. A lo largo de ellos se pretende exponer de un modo claro, conciso y detallado los aspectos sobre él que se consideran relevantes. La siguiente lista describe someramente cuál es el contenido que se pretende transmitir en cada parte:

- En este capítulo, la **Introducción**, mostramos un resumen de las investigaciones y problemas que han motivado la realización de esta investigación. También se han presentado las principales ideas en las que se basa el sistema implementado que se expone en el trabajo, y se ha descrito de manera general este sistema.

- En el Capítulo 2, **Trabajo previo**, se presenta el estado del arte de las investigaciones y tecnologías sobre las que se ha construido el sistema de generación de historias.

- El Capítulo 3, **Sistema de generación de historias**, explicamos con todo detalle la aplicación que se ha desarrollado, junto con las ideas y aportaciones que se han realizado.

- En **Discusión y propuestas**, el Capítulo 4, se discuten los principales puntos de nuestra aportación en relación con el trabajo previo, poniendo de manifiesto cuáles son las ventajas e inconvenientes del programa realizado y las ideas aportadas en comparación con el estado del arte de las materias en las que se apoya esta investigación.

- Finalmente, **Conclusiones y trabajo futuro** (Capítulo 5), resume la investigación realizada, resaltando los principales puntos, y ofrece una lista de tareas y futuras ampliaciones posibles que son susceptibles

de ser llevadas a cabo en la investigación posterior que sigue a esta investigación.

Tras los capítulos y la bibliografía, se han añadido una serie de apéndices sobre diferentes puntos de esta investigación, los siguientes:

- El Apéndice A, **Publicaciones**, contiene las publicaciones que se han realizado a lo largo de y en relación con este trabajo.
- En **Ejemplo de XML de entrada** (Apéndice B) mostramos un ejemplo de la utilidad del sistema, mostrando un fragmento de un archivo de entrada que proviene de una aplicación de simulación multiagente, y el texto generado correspondiente.
- En el Apéndice C, **Entrada para el fragmento del Menón**, mostramos la descripción estructurada en forma de archivo de entrada de nuestro sistema para la generación de un fragmento del diálogo platónico de *Menón*.
- Finalmente, el Apéndice D, **Entrada para el ejemplo de los barcos autónomos**, muestra la entrada correspondiente a una ejecución de barcos autónomos a escala de manera simulada, cuya salida también narramos.

Capítulo 2

Trabajo previo

A lo largo de este capítulo se ofrece una visión general de las técnicas, herramientas e investigaciones que han dado lugar a este trabajo. Sobre los aspectos más relevantes de estas ideas en relación con la investigación que se presenta en el capítulo siguiente, a lo largo de este documento, y en especial en el Capítulo 4, se hará un análisis comparativo y crítico.

Esta recopilación sobre el estado del arte está dividida en cinco secciones principales se cuentan las técnicas, ideas, y trabajos más importantes sobre los que se apoya esta investigación. En la primera sección se habla de Generación de Lenguaje Natural, y en la segunda trataremos los aspectos más importantes de una subdisciplina de la GLN, la Generación de Historias en Lenguaje Natural.

En la Sección 2.3 estudiamos el concepto y las aportaciones más importantes sobre Creatividad Computacional. En las secciones 2.4 y 2.5 hablamos de dos tecnologías sobre las que se ha implementado el prototipo para la investigación: XML y Ontologías, respectivamente.

2.1. Generación de Lenguaje Natural

La Generación de Lenguaje Natural (GLN) es un área de investigación proveniente de la Inteligencia Artificial y de la Lingüística Computacional, y está orientada al estudio de sistemas capaces de generar textos de forma automática en un idioma humano, como el español o el inglés. Típicamente, la GLN parte de una representación estructurada no lingüística, y construye textos como informes, historias, partes de diálogos, mensajes de ayuda, y demás.

La GLN es una de los dos procesos principales en el campo del Procesamiento de Lenguaje Natural (PLN). Éste se divide, según una taxonomía típica, en GLN y Comprensión del Lenguaje Natural (CLN), que puede ser considerada, desde un punto de vista funcional, como la tarea inversa de la GLN, ya que la CLN está orientada a recibir mensajes en un

idioma natural, de parte de un humano, y traducirlos a estructuras que pueden ser interpretadas o "entendidas" por una máquina, y la GLN comprende aquellos procesos inversos, es decir, de una representación computacional de un conjunto de datos, tratarlos de tal modo que se generen mensajes a partir de ellos en un lenguaje humano.

2.1.1. Breve historia de la Generación de Lenguaje Natural

Los primeros trabajos relacionados con la GLN comenzaron entre los años cincuenta y sesenta del siglo XX. Básicamente eran primeras investigaciones dedicadas al estudio de la traducción automática. No fueron hasta los años setenta las primeras aportaciones que realmente separaron la GLN de la CLN, con las investigaciones de [Gol75, Dav78]. En estos trabajos se puso de manifiesto que, en contra de lo que algunas corrientes científicas postulaban, la GLN no podía ser tratada como el proceso inverso de la CLN, puesto que estas dos disciplinas tienen problemas y características no complementarias o inversas.

Ya en los años ochenta, la investigación sobre la GLN experimentó un avance notable. Los trabajos doctorales de McKeown y Appelt [McK85, App85] por ejemplo, ejercieron una influencia notable en las técnicas e ideas que después iban a ser desarrolladas. Aparecieron los primeros congresos sobre GLN (International Workshop on Natural Language Generation, 1983), y cobró fuerza la tendencia de crear sistemas particulares sobre áreas más específicas de la GLN, dejando atrás los intentos de desarrollo de grandes sistemas de programación monolíticos que pretendían resolver muchos problemas de una manera acoplada.

Durante los años noventa y actualmente, la GLN sigue muchas de las tendencias y estudios iniciados en los años ochenta, y ha aparecido un interés grande en la fusión de técnica de generación textual de contenido con otras posible representaciones de la realidad, como gráficos, sonido y animaciones. La creciente capacidad de cálculo y almacenamiento de los ordenadores modernos, además, ha facilitado y propiciado muchos avances que antes, por las restricciones de las plataformas de computación, no eran posibles.

2.1.2. Etapas de la Generación de Lenguaje Natural

La generación de texto en lenguaje natural, generalmente, está dividida en varias etapas secuenciales que, sucesivamente, refinan el contenido original hasta darle la forma necesaria de texto en un idioma natural dado [RD00]. De una manera simplificada, podemos decir que el proceso de generación de lenguaje natural típico se realiza aplicando diferentes operaciones complejas, que no han de ser sucesivas necesariamente, ni estar desacopladas. Son, someramente, las siguientes:

- Planificación del texto, que consiste en la elección del contenido

que va a ser traducido a lenguaje natural, a partir del conocimiento de entrada y el objetivo del discurso. Comprende dos operaciones:

  - Determinación del contenido, o la decisión de qué elementos de la historia han de ser contados, y cuáles han de ser omitidos. Esta etapa es muy dependiente del dominio, y no es posible una gran generalización.
  - Planificación del discurso, que consiste en el establecimiento del orden en el que se cuentan las partes de la historia, y cómo se enlazan entre sí.

- Planificación de las frases, también conocida como Microplanificación, que recibe el contenido filtrado y ordenado que conformará la información que transmitir de la historia, y procesa los mensajes para crear estructuras de frases más cercanas a como las creamos los humanos. Agrupa tres tipos de procesos:

  - Agregación, que consiste en las operaciones de unir frases separadas con significados que las hacen poder estar escritas en una única elocución.
  - Lexicalización, etapa en la cual se decide qué palabras y frases específicas del lenguaje objeto van a ser usadas para describir los hechos.
  - Generación de expresiones de referencia, o cómo describir cada hecho de la historia teniendo en cuenta su colocación dentro del texto global.

- Realización lingüística, que, finalmente, enlaza el discurso haciendo la flexión gramatical de las partes, y establece una ortografía correcta. Engloba dos tipos de operaciones:

  - Realización sintáctica y morfológica, atendiendo a la flexión de las reglas gramaticales.
  - Realización ortográfica, o el ajuste de la escritura del texto a las reglas ortográficas de la lengua natural en la que se genera el mensaje final.

El uso más común de la generación de lenguaje natural es crear sistemas que presenten la información al usuario en una representación fácil de comprender. Internamente, estos sistemas usan representaciones que son más directas de manipular, tales como horarios de líneas aéreas, bases de conocimiento de sistemas expertos, simulaciones de sistemas físicos... En muchos casos, sin embargo, estas representaciones necesitan una gran cantidad de experiencia para ser interpretadas, y por ello es necesario presentar la información al usuario no experto de una manera más clara. Por ejemplo, se han usado técnicas de GLN para:

- Generar predicciones meteorológicas textuales a partir de representaciones en mapas gráficos, como en [GDK94].
- Resumir datos estadísticos extraídos de una base de datos, como en [IKK+92].
- Explicar información médica de una forma amigable para el paciente, como en [CBJ95, BMF+95].
- Describir una cadena de razonamiento llevada a cabo por un sistema experto, como en [Swa83].
- Producir respuestas a preguntas sobre un objeto descrito en una base de conocimiento, como en [RML95].
- Generar textos literarios, como en [CL01].

En este artículo, el trabajo aportado se centra en las dos primeras etapas, que se engloban en las operaciones generales denominadas Planificación del contenido. Aparte de nuestra propuesta, podemos encontrar diferentes aproximaciones al tema en trabajos relacionados con la Teoría de Estructura Retórica [Hov93, MT88], o Schemas [McK85]. En ambos casos, las reglas que gobiernan el comportamiento del sistema han de ser escritas a mano, y son dependientes del dominio de trabajo y de la aplicación, por lo que deben ser, en cada caso, reescritas para cada migración de los mismos.

A continuación se añade algunas ideas más específicas sobre las operaciones de GLN más relacionadas con este trabajo de investigación: la Determinación del contenido y la Planificación del discurso. Además, puesto que el prototipo del sistema que se presenta realiza también operaciones de Realización superficial, damos también algunas referencias sobre este tema.

2.1.3. Determinación del contenido

La Determinación del contenido agrupa todas aquellas operaciones que se dedican a decidir qué ha de ser contado y qué no. En el ámbito de esta investigación, puesto que se dispone de una gran cantidad de datos de entrada, es uno de los aspectos más importantes.

En algunos sistemas de generación de lenguaje natural, la Determinación del contenido no ha de ser realizada, ya que los mismos sistemas proveen en la entrada de datos el conjunto de los mismos que han de ser descritos o narrados. No obstante, este no es un caso típico de los problemas reales de la GLN. En [McD99] se describen estas dos alternativas como de tipo PUSH y PULL, respectivamente.

Decidir qué contenido aparecerá en el mensaje global final no es una tarea ni mucho menos trivial, y es aceptado en el campo de la GLN que esta

operación es en gran medida dependiente del dominio de aplicación. Por esto, no es posible crear reglas globales que modelen el comportamiento genérico de un determinador del contenido. A pesar de que, intuitivamente, reglas típicas como *relatar sólo lo relevante* o *evitar hechos fácilmente inferibles* suelen ser aplicadas, la solución a estas reglas suele estar, de nuevo, ligada al dominio en el que se genera el texto en lenguaje natural.

Aparte de estas ideas, la DETERMINACIÓN DEL CONTENIDO depende, a grandes rasgos, de los siguientes factores:

- **Los objetivos de la comunicación**. Claramente, lo que se pretenda decir ha de ser la principal guía de la decisión sobre el contenido que ha de ser transmitido en la comunicación que se produce en la generación del texto.
- **Las características del lector**. Teniendo en cuenta el modelo implícito o explícito que dispongamos del lector u oyente del mensaje en lenguaje natural que queremos construir, habremos de elegir unos hechos u otros. Por ejemplo, sería interesante, en un sistema de diálogo, tener en cuenta lo que el interlocutor de la máquina ya sabe, y responder en consecuencia.
- **Restricciones en la salida**. Si se especifica que el texto final debe contener una serie de datos, estos, obligatoriamente, habrán de ser incluidos en el subconjunto que decida la fase de determinación.
- **El origen de la información** es fundamental e influye en gran medida en el contenido que va a ser elegido para aparecer en el texto final, ya que la información de la que dispongamos, evidentemente, restringe las soluciones posibles.

2.1.4. Planificación del discurso

Existen, hoy en día, una gran cantidad de alternativas disponibles para llevar a cabo la tarea de PLANIFICACIÓN DEL DISCURSO, o cómo hilar los hechos de una historia de modo que tenga ésta sentido y coherencia, y pueda transmitir el mensaje o historia al lector de la manera que éste pueda entenderlo más fácilmente.

Como es evidente para un humano, un mensaje en texto no consta simplemente de una lista de hechos sin ningún orden ni relación entre ellos que añaden información sobre algún dominio en particular. Si un lector lee los párrafos de un capítulo determinado de un libro de manera desordenada, muy probablemente no aprehenderá la información que el escritor deseaba transmitir. Es, por tanto, necesario imponer un orden al texto. Hasta en las aproximaciones más simples, por ejemplo, de las fábulas, se sigue un esquema básico de `situación`–`problema`–`moraleja`, aunque, por supuesto, la mayoría de los textos están articulados con estructuras más complejas.

El análisis más trivial que se puede hacer de la estructura y el orden de un texto es la secuencia lineal de los hechos que lo componen. Sin embargo, esto no es usualmente suficiente. En general, los textos poseen un hilo más intrincado y complejo que la simple lista, estando formado muy a menudo por estructuras de árbol. En ellas, los hechos de unas ramas tienen todo tipo de relaciones con, potencialmente, cualquier parte del texto, aunque explícitamente sólo queden reflejadas algunas.

Estructuras de árbol muy claras las presentan los textos de, por ejemplo, trabajos como el presente, en el que la información está mostrada como una lista de hechos, sino que los capítulos, secciones y párrafos dan profundidad, agrupación y anidamiento al texto; no sólo desde un punto de vista de la maquetación, sino, y sobre todo, para dar algo más de semántica a la información que se presenta. Las diferentes ramas en un árbol estructural del texto permiten, por tanto, crear relaciones de tipo retórico, tal como se explica, por ejemplo, en la RST, o Teoría de la Estructura Retórica [MT88] .

La mayoría de las tendencias sobre la planificación del discurso están basadas en el objetivo. Esta perspectiva facilita mucho la comprensión del acto comunicativo del hombre, tras cuyo análisis es posible crear máquinas que puedan emular al ser humano en este campo. La orientación al objetivo consiste en descubrir o definir qué quiere el escritor o hablante que el lector u oyente sepan, y cómo actúa en consecuencia a eso para lograr transmitir la información que quiere.

Es fundamental notar que la Planificación del discurso y la Determinación del contenido son tareas que, muy a menudo, aparecen acopladas en su propia ejecución y algoritmos. Esto es debido a que es posible que lo que se deba o no contar dependa de la posible ordenación que le demos al texto, y viceversa. Para un ejemplo simple, si pretendemos organizar el texto en capítulos, es posible que sea necesario incluir información de cabecera como introducción de cada uno de ellos.

Antecedentes teóricos de la planificación del discurso

Antes de la generación computacional de textos desde el punto de vista de la estructura del discurso que presentan se han dado una serie de estudios relacionados con la Lingüística que pretenden estudiar y analizar la forma en la que los humanos nos comunicamos. Estos estudios fueron iniciados desde muy antiguo (Aristóteles ya reconoció que los fragmentos de las comunicaciones, desde el punto de vista de la intención hacia el oyente, se enlazan con sólo un pequeño conjunto de posibles relaciones). Básicamente, podemos decir que hay dos tendencias principales: la *formalista* y la *funcionalista* [Hov93].

La teoría *formalista* [Kam81] postula que los discursos narrativos contienen una estructura interna, y que cada elemento de esa estructura encierra

una serie de unidades que están relacionadas entre sí. Estas estructuras suelen tener algunos puntos no muy bien apoyados cuando se analizan las partes semánticas de estas estructuras (lo que realmente quieren decir o transmitir). La Teoría de Representación del Discurso [Kam81] es la más influyente. Existen otras, relacionadas, por ejemplo, con las estructuras de punto de vista de todo el discurso (macro-discurso) como [Dij72], o gramáticas de historias, como en [Rum72].

La teoría *funcionalista* también admite que los discursos poseen una estructura interna determinada, pero se enfoca más al estudio de que esta estructura está definida por el propósito u objetivo de la comunicación [Lev79]. Sobre este enfoque existen muchas alternativas sobre conjuntos de relaciones que se pueden hallar entre las diferentes sentencias, como podemos encontrar en [Hob78, Hob79, Gri75, She26, Dah88, MT88, Mar92], por nombrar algunas. El estudio de éstas está más allá del enfoque de este estudio sobre los antecedentes teóricos, pero es importante notar que el presente trabajo se basa en uno de estos conjuntos de relaciones, la Teoría de la estructura Retórica, de [MT88].

Ambas teorías, poco a poco, se unifican en una sola, ya que no son excluyentes entre sí. En [GS86] ya se puede ver una teoría del discurso que las aglutina en una sola. Esta teoría expone una posible análisis triple paralelo e el que pueden ponerse de manifiesto las teorías *formalista* de la estructura de las elocuciones, la *funcionalista* de las intenciones del locutor, y una más que hace referencia a aquellos objetos que están en la "memoria del discurso", es decir, que pueden ser referenciados en su desarrollo.

Investigaciones y trabajos previos de la planificación del discurso

Las primeras aproximaciones en planificación del discurso que se llevaron a cabo simplemente ignoraban todas estas ideas teóricas sobre cómo montar una estructura textual coherente, y se limitaban a "encontrar y consumir" hechos de una base de conocimiento de manera que acabasen formando parte del discurso final. Entre estos trabajos podemos encontrar KDS [MT81]. TaleSpin [Mee76] y Proteus [Dav78] funcionaban según la organización de la semántica del dominio en el que trabajaban. Algunos también basaban su funcionamiento, principalmente, en aplicar reglas y plantillas directamente sobre los datos, de manera que tenían un control total sobre el algoritmo de generación, poniendo de manifiesto, de nuevo, lo muy dependientes del dominio que pueden llegar a ser los planificadores de contenido [BFM80, Syc87].

Construcción de arriba hacia abajo

La mayoría de los planificadores de texto clásicos asumen que los textos generados, desde un punto de vista estructural, tiene forma de árbol. Por

esto se afronta la planificación del discurso como una aplicación en búsqueda de una serie de operadores que crean un espacio de estados de "arriba hacia abajo", es decir, con búsquedas jerárquicas. Algunos de estos trabajos se pueden encontrar en [Hov93, MP91, MS91, Caw90, May90]. Estas búsquedas jerárquicas tienen algunos problemas, como discutimos en la Sección 4.5.1.

También la aplicación de SCHEMAS (Sección 2.1.6) en la realización del discurso textual puede ser considerada como una técnica de arriba hacia abajo.

Construcción de abajo hacia arriba

La otra posibilidad existente es el enfoque que se presenta en [Mar97b]. En este trabajo se nos ofrece una alternativa a las búsquedas de "arriba hacia abajo", invirtiendo el orden. En vez de partir de una estructura que va reduciéndose paulatinamente hasta encontrar mensajes en una base de conocimiento que pueden ser incluidos en la historia, en este tipo de construcción se aboga por partir de los mensajes de los que se dispone, los *átomos* de información, e ir agrupándolos en estructuras cada vez más complejas hasta formar un discurso coherente.

El proceso se basa en conocer qué relaciones (por ejemplo, relaciones de la RST) existen entre cada par de elementos de semántica a cierto nivel, empezando por los elementos más básicos de los que se dispone. Una vez que se han identificado estas posibles relaciones, se usa una función heurística para decidir cuál de los grupos de mensajes unidos por una relación determinada de todos los candidatos es el elegido para formar parte de una estructura superior. Este paso, en el algoritmo, se repite hasta que no quedan elementos sin agrupar.

De este modo se obtiene una historia coherentemente contada y que incluye todos los mensajes que quieren ordenarse. No obstante, este tipo de algoritmo tiene el inconveniente principal de que encontrar la función heurística es demasiado dependiente del dominio, y que, incluso con eso, en ocasiones es extremadamente difícil de de obtener.

2.1.5. Creación del texto final

Tras la creación de una estructura textual correcta que representa las ideas que quieren transmitirse, ya filtradas y ordenadas según una jerarquía determinada, es posible realizar una traducción de los datos a un lenguaje natural. Esta parte de las operaciones se conoce como planificación de las frases (SENTENCE PLANNING) y realización superficial (SURFACE REALIZATION).

Durante la planificación de las frases se llevan a cabo las tareas de LEXICALIZACIÓN, AGREGACIÓN y GENERACIÓN DE EXPRESIONES DE REFERENCIA. La primera consiste en elegir un conjunto determinado de palabras y

frases que van a ser sustituciones de los conceptos con los que se ha realizado la planificación del discurso. [Cah98] establece que en realidad hay una diferencia importante entre la "realización léxica", que se refiere a la conversión de conceptos en etiquetas léxicas, y con la que tenemos que disponer de etiquetas para todos los conceptos; y la "elección léxica", que se encarga de escoger la mejor opción de entre las alternativas que representa el mismo concepto, en la que los recursos léxicos deben proporcionar conocimiento suficiente para llevar a cabo la traducción de información.

Según [RM99], la AGREGACIÓN, que consiste en la unión de varios mensajes en uno solo para que la legibilidad sea mejor, puede dividirse en seis tipos: *conceptual*, *de discurso*, *semántica*, *sintáctica*, *léxica* y *referencial*. No obstante, esta diferenciación no es formal ni está totalmente clara, y podemos encontrar tipos de agregación con los que cabría una clasificación en diferentes tipos.

La GENERACIÓN DE EXPRESIONES DE REFERENCIA es un conjunto de operaciones e ideas que están orientadas a elegir las palabras adecuadas para referirse a las diferentes entidades de un texto concreto según las diferentes partes de discurso. Esta fase requiere un conocimiento importante del contexto en el que se va a aplicar cada expresión de referencia. Es necesario, siguiendo una regla usual presente en [RD92] que la referencia que se expresa debe dejar claro cuál es el referente de manera unívoca.

Finalmente, una vez que tenemos creado el contenido que almacena la representación de las frases, es posible traducirlo, finalmente a un lenguaje natural. Es en esta etapa de la GLN donde se aplican las reglan ortográficas y morfológicas apropiadas para que se pueda escribir un texto que siga las estructuras que se han producido. Existen varios sistemas que realizan esta acción, como [Elh93, ER96].

2.1.6. Schemas

Según psicología del lenguaje [BIR04] los seres humanos, cuando queremos comunicarnos entre nosotros, usamos un conjunto claro de esquemas de comunicación a muchos niveles. Las frases que empleamos tienen una estructura clara dependiente, claro, del lenguaje en el que se expresan. Los escritos que creamos, también sigue una estructura típica (`título`–`introducción`–`desarrollo`–`conclusiones`, por ejemplo).

El sistema TEXT [McK85] es uno de los primeros sistemas que tiene realmente en cuenta la estructura del textos. McKeown propone unas estructuras denominadas SCHEMAS que pueden dirigir la generación de texto, conteniendo cada esquema más esquemas anidados, y cada uno de estos está definido en función de un predicado retórico.

Siguiendo estas ideas, esta técnica consiste en aplicar patrones fijos de estructuras para la creación de discursos en lenguaje natural a partir de contenido almacenado en un ordenador de manera conceptual. Mediante la

aplicación de esquemas de un modo anidado se consigue un árbol narrativo que representa no sólo el orden de los hechos, sino también los grupos de los mismos que conforman párrafos, capítulos o incluso ideas. Esto depende de los esquemas que se traten. En la Figura 2.1 podemos ver un ejemplo de esos esquemas. Vemos que son perfectamente programables en un lenguaje de programación cualquiera.

```
MensajeIntroduccion():
    Elegir elementos de introducción
    Si tenemos más de uno:
        Crear una relación de conjunción entre ellos
        Devolver un párrafo con todos los mensajes y
        la conjunción
    Si no:
        Devolver MensajeNoHayIntroducción()
```

Figura 2.1: Ejemplo de esquemas para Generación de Lenguaje Natural.

Los Schemas son una herramienta útil y sencilla de usar, además de implementable en cualquier lenguaje Turing-completo. Por tanto, han sido extensamente utilizados.

Los esquemas crean estructuras computacionales que tienen mucho parecido con las Gramáticas Incontextuales. En el apartado 2.1.6 damos una pequeña introducción a ellas. Esta estructura que se crea, por tanto, nos hace poder profundizar en un árbol de generación de textos de modo que conseguimos, tal y como se hace con estas gramáticas, una anidación de contextos que nos ofrece la potencia de crear "subapartados" o incluso "subhistorias" dentro de las historias que se van a generar.

Gramáticas independientes del contexto

Hemos comentado anteriormente que los Schemas se programan en una estructura que tiene características de gramática incontextual, o gramática independiente del contexto [Sip97]. En esta sección vamos a hacer un breve resumen sobre estas gramáticas.

Las gramáticas incontextuales son una gramáticas formales que permiten expresar producciones de la forma:

$$V \longrightarrow w \tag{2.1}$$

donde V se llama *símbolo no terminal*, y v *símbolo terminal*, y quiere decir que V puede ser traducido por v, *independientemente del contexto de* V. Estas gramáticas tienen la expresividad necesaria para definir las estructuras lingüísticas que conforman los lenguajes de programación usuales.

Formalmente las GRAMÁTICAS INCONTEXTUALES se definen por una 4-tupla del tipo siguiente:

$$G = (V_t, V_n, P, S) \tag{2.2}$$

donde:

- V_t es un conjunto finito de terminales.
- V_n es un conjunto finito de no-terminales.
- P es un conjunto de reglas de producción, como en la Ecuación 2.1.
- S es el símbolo de comienzo de la gramática.

2.1.7. Teoría de la Estructura Retórica (RST)

La RST[1], o TEORÍA DE LA ESTRUCTURA RETÓRICA es una teoría que ofrece un mecanismo para explicar la coherencia textual, independientemente de las formas léxicas y gramaticales del texto [MT88, MT92]. La RST expone que los textos con los que los humanos se comunican diariamente no están simplemente compuestos de frases aleatorias o desorganizadas entre sí, sino que cada parte de un texto que es considerado *coherente* tienen un papel evidente, una intención de transmisión de significado concreta y en relación con las demás partes del texto. En cuanto a la *coherencia*, la definición de la misma con la que trabaja la RST dice que un texto coherente es aquel en el que no existen lagunas (ideas importantes no explicadas o relacionadas) ni secuencias ilógicas. Es importante hacer énfasis en la idea de que la RST describe textos, no la manera de crearlos o analizarlos.

La RST tiene su origen en la generación automática de textos [RD00, Tab06]. Bill Mann, Sandy Thompson y Christian Matthiessen observaron a principios de los años ochenta, mientras trabajaban en la elaboración de textos mediante computadora, que no existía ninguna teoría que explicara la función de cada parte de un texto dentro de un mensaje global. Por esta razón desarrollaron la RST, para poder disponer de un conjunto de ideas que sirviese como base teórica para la generación de textos. Hoy en día, la RST se emplea no sólo dentro de los campos de la teoría de la computación en relación con el lenguaje, sino que es una teoría aceptada y estudiada dentro de la lingüística como disciplina general.

El grueso de la aportación de esta teoría consiste en una lista de ESTRUCTURAS, o unidades fundamentales de significado que son observables por un lector u oyente. La RST contempla dos niveles de estas estructuras. Por un lado, las conocidas como RELACIONES, y por otro, los ESQUEMAS, que definen de qué manera es posible construir y organizar los textos según estas relaciones.

[1] Rhetorical Structure Theory

Las RELACIONES enlazan unidades semánticas con las reglas de una estructura dada, y mediante una relación que especifica cuál es la función de unas con respectos a las otras. Por ejemplo, en la frase "Platón escribió diálogos, era filósofo y quería enseñar sus ideas", tenemos que existen dos partes bien diferenciadas en el mensaje, que no tienen que ver con la estructura sintáctica que proporciona la gramática del español: "Platón escribió diálogos" y "era filósofo y quería enseñar sus ideas". Estas dos partes son unidades, y existe un relación de ELABORACIÓN entre ellas dos, como explicamos más adelante. Los ESQUEMAS, por su parte, establecen cómo se construyen elementos a partir de estas relaciones.

La RST ofrece conjuntos personalizables de relaciones que unen los diferentes segmentos del texto (generalmente, estos segmentos son frases o párrafos). Mediante estas relaciones, que pueden ser de diferentes tipos, se crea un árbol que da coherencia y organización al conjunto de información. El conjunto de estas relaciones ha ido evolucionando según se ha ido investigando en la RST. A fecha de este trabajo, las relaciones son las que exponemos en las Tablas 2.1, 2.2, 2.3 y 2.4; respectivamente, relaciones de PRESENTACIÓN, de CONTENIDO (I y II) y MULTINUCLEARES.

N es el núcleo, S el satélite, A el autor o autora (escritor/a o hablante) y L el lector o lectora (también oyente). Para mayor brevedad, en las definiciones, N y S se refieren a las situaciones que N y S representan; N y S nunca se refieren al texto de N o S. Situación es un término amplio, por el que se entiende proposiciones o creencias, acciones realizadas o no, deseos de actuar y la aprobación para que otra persona actúe. De igual manera, actitud positiva es un término de actitud que, de manera amplia, cubre creencias, aprobación de ideas, deseo de actuar, y aprobación para que otra persona actúe, todos ellos obviamente positivos. Los términos actitud positiva, creencia (y sus derivados) y verosímil son términos en una escala, no términos binarios.

Las definiciones (las condiciones que el observador u observadora debe constatar), igual que los nombres, están organizadas en tablas que se dividen en relaciones de presentación, relaciones de contenido y relaciones multinucleares. Los nombres de estas relaciones en español están tomados de [Ber95].

Las relaciones de PRESENTACIÓN y de CONTENIDO son consideradas como relaciones de tipo NÚCLEO–SATÉLITE, ya que en ellas existe una parte que es la principal, desde el punto de vista semántico, y otra u otras que tienen una función hacia ella. Este tipo de relaciones retóricas es muy habitual, y se suele dar entre elementos adyacentes del texto. Sin embargo, es importante tener en cuenta que no existe, habiendo hecho el análisis de cómo nos expresamos con textos, un orden definido ni obligatorio en este tipo de relaciones.

Las relaciones MULTINUCLEARES son aquellas que están compuestas por elementos de significado que están al mismo nivel entre sí. Es decir, no existe un núcleo que contenga el significado principal de la proposición, sino que

Nombre de la Relación	Condiciones en S o N, individualmente	Condiciones en N + S	Intención de A
Antítesis	en N: A tiene una actitud positiva hacia N	N y S se encuentran en contraste (véase la relación Contraste), dada la incompatibilidad que resulta del contraste, no es posible tener una actitud positiva hacia ambas situaciones, la comprensión de S y la incompatibilidad aumenta la actitud positiva de L hacia N	Aumenta la actitud positiva de L hacia N
Capacitación	en N: presenta una acción por parte de L (que incluye la aceptación de una oferta), no realizada en el marco contextual de N	La comprensión de S por parte de L aumenta la capacidad de L para llevar a cabo la acción	Aumenta la capacidad de L para llevar a cabo la acción
Concesión	en N: A tiene una actitud positiva hacia N en S: A no afirma que S no es cierto	A reconoce una (posible) incompatibilidad entre N y S; el reconocimiento de la compatibilidad entre N y S aumenta la actitud positiva de L hacia N	Aumenta la actitud positiva de L hacia N
Evidencia	en N: L podría no creer N de manera satisfactoria para A en S: L acepta S o lo encuentra creíble	La comprensión de S por parte de L aumenta la aceptación de N por parte de L	Aumenta la aceptación de N por parte de L
Fondo	en N: L no entenderá N completamente antes de leer el texto de S	S aumenta la capacidad de L para entender un elemento en N	Aumenta la capacidad de L para entender N
Justificación	ninguna	La comprensión de S por parte de L aumenta su inclinación a aceptar que A presente N	Aumenta la inclinación de L a aceptar que A presente N
Motivación	en N: N es una acción en la que L es el actor (incluye la aceptación de una oferta), no realizada con respecto al marco contextual de N	La comprensión de S por parte de L aumenta su deseo de llevar a cabo la acción presentada en N	Aumenta el deseo de L de llevar a cabo la acción presentada en N
Preparación	ninguna	S precede a N en el texto; S hace que L se sienta más preparado, interesado u orientado para leer N	L se siente más preparado, interesado u orientado para leer N

Tabla 2.1: Relaciones RST de presentación.

cualquier elemento del conjunto tiene la misma función o importancia que cualquier otro. Las copulaciones disyuntivas o conjuntivas en español son un buen ejemplo de lo que queremos decir con relaciones MULTINUCLEARES.

Las RST dispone de un conjunto de herramientas que pueden ser descargadas de forma gratuita de Internet mediantes las que se puede dotar a un texto de esta estructura retórica. Son aplicaciones gráficas que permiten, mediante el uso del ratón, unir fragmentos de textos de tal modo que, de manera resultante, tengamos un árbol documental en el que queden clara y formalmente reflejadas las relaciones entre los diferentes constituyentes del texto. Una de las mejores es RSTTool [O'D00]. Esta aplicación gráfica puede importar textos planos, dividir automáticamente el texto en partes, y dejar al usuario que genere la estructura del texto. Después, es capaz de exportar archivos en XML con el contenido de esta estructura retórica de los textos. En la Figura 2.2 podemos ver una captura de pantalla de esta

Nombre de la Relación	**Condiciones en S o N, individualmente**	**Condiciones en N + S**	**Intención de A**
Reformulación	ninguna	en N + S: S reformula N, siendo S y N de tamaño similar; N es más importante para los propósitos de A que S	L reconoce S como una reformulación de N
Resumen	en N: N debe estar constituido por más de una unidad	S presenta una reformulación del contenido de N, más reducida	L reconoce S como una breve reformulación de N
Alternativa	en N: es una situación no realizada en S: S es una situación no realizada	La realización de N impide la realización de S	L reconoce que la realización de N impide la realización de S
Causa Involuntaria	en N: N no es una acción voluntaria	S causó N, por medios diferentes a los que motivan una acción voluntaria; sin la presentación de S, L podría no saber la causa de la situación; la presentación de N es más importante que la de S para los fines de A al presentar la combinación N-S	R reconoce S como causa de N
Causa Voluntaria	en N: N es una acción voluntaria o una situación que podría haber surgido de una acción voluntaria	S podría haber llevado al agente de la acción voluntaria en N a realizarla; sin la presentación de S, L podría no considerar la acción motivada; N es más importante que S para los fines de A al presentar la combinación N-S	L reconoce que S es la causa de la acción voluntaria en N
Circunstancia	en S: S no se encuentra sin realizar	S establece un marco para el tema principal, dentro del cual L ha de interpretar N	L reconoce que S proporciona el marco para la interpretación de N
Condición	en S: S presenta una situación hipotética, futura, o aún no realizada (con relación al marco contextual de S)	La realización de N depende de la realización de S	L comprende cómo la realización de N depende de la realización de S
Condición Inversa	ninguna	S afecta la realización de N; N se llevará a cabo solo si S no se lleva a cabo	L reconoce que N se llevará a cabo solo si S no se lleva a cabo
Elaboración	ninguna	S presenta detalles sobre la situación o algún elemento en N o accesible en N mediante una de las inferencias que se presentan a continuación. En la lista, N se refiere a la primera parte del par, y S a la segunda, según la lista: *conjunto :: miembro*, *abstracto :: ejemplo*, *todo :: parte*, *proceso :: paso*, *objeto :: atributo*, *generalización :: específico*.	L reconoce que la situación presentada en S proporciona detalles para N. L identifica el elemento para el que se han proporcionado los detalles
Evaluación	ninguna	en N + S: S refiere N al grado de actitud positiva por parte de A con respecto a N.	L reconoce que S afirma N y reconoce el valor que se le ha asignado

Tabla 2.2: Relaciones RST de contenido (I).

herramienta en funcionamiento.

Hay aún muchos problemas que no están resueltos, y que son el principal punto de investigación actual en el campo de esta teoría. Entre ellos, podemos destacar los problemas sobre la diferencia entre las lenguas (no todos los idiomas construyen textos con los mismos esquemas), la comprensión de cómo el hombre hace el análisis RST para poder aplicarlo a las máquinas, los problemas que surgen cuando se aplica a formatos más complejos que los del simple monólogo textual, y otros[2]. Existen, además, grandes relaciones entre la RST y otras áreas de la lingüística.

[2]http://www.sfu.ca/rst/08spanish/areas.html

Nombre de la Relación	Condiciones en S o N, individualmente	Condiciones en N + S	Intención de A
Interpretación	ninguna	en N + S: refiere N a un marco de ideas no incluido en N y que no tiene relación con la actitud positiva de A	L reconoce que S refiere N a un marco de idas no incluido en el contenido presentado en N
Método	en N: una actividad	S presenta un método o instrumento que puede hacer posible la realización de N	L reconoce que el método o instrumento en S puede hacer posible la realización de N
No-condicional	en S: S podría afectar la realización de N	N no depende de S	L reconoce que N no depende de S
Resultado Involuntario	en S: S no es una acción voluntaria	N causó S; la presentación de N es más importante que la de S para los fines de A al presentar la combinación N-S	L reconoce que N podría haber causado la situación en S
Resultado Voluntario	en S: S es una acción voluntaria o una situación que podría haber surgido de una acción voluntaria	N podría haber causado S; la presentación de N es más importante que S para los fines de A.	L reconoce que N podría ser causa de la acción o situación en S
Propósito	en N: N es una actividad; en S: S es una situación no realizada	S se llevará a cabo mediante la actividad en N	L reconoce que la actividad en N tiene como propósito llevar a cabo S
Solución	en S: S presenta un problema	N es una solución al problema presentado en S	L reconoce N como solución al problema presentado en S

Tabla 2.3: Relaciones RST de contenido (II).

Nombre de la Relación	Condiciones en cada par de N	Intención de A
Contraste	no más de dos núcleos; las situaciones en estos núcleos (a) se entienden como la misma en muchos aspectos, (b) se entienden como diferentes en algunos aspectos, y (c) se comparan con respecto a una o más de estas diferencias	L reconoce la posibilidad de comparación y la(s) diferencia(s) presentadas en la comparación
Lista	Un elemento comparable a otros y unido al otro N mediante la relación Lista	L reconoce la comparación de los elementos en la lista
Reformulación Multinuclear	Un elemento es una repetición de otro al que se encuentra unido; los elementos son de importancia similar con respecto a los fines de A	L reconoce la repetición de los elementos unidos
Secuencia	Existe una relación de sucesión entre las situaciones presentadas en los núcleos	L reconoce la sucesión de relaciones entre los núcleos
Unión	ninguna	ninguna

Tabla 2.4: Relaciones RST multinucleares.

Aplicaciones de la RST

Existe una gran cantidad de aplicaciones concretas que hacen uso de la Teoría de la Estructura Retórica, principalmente porque esta teoría

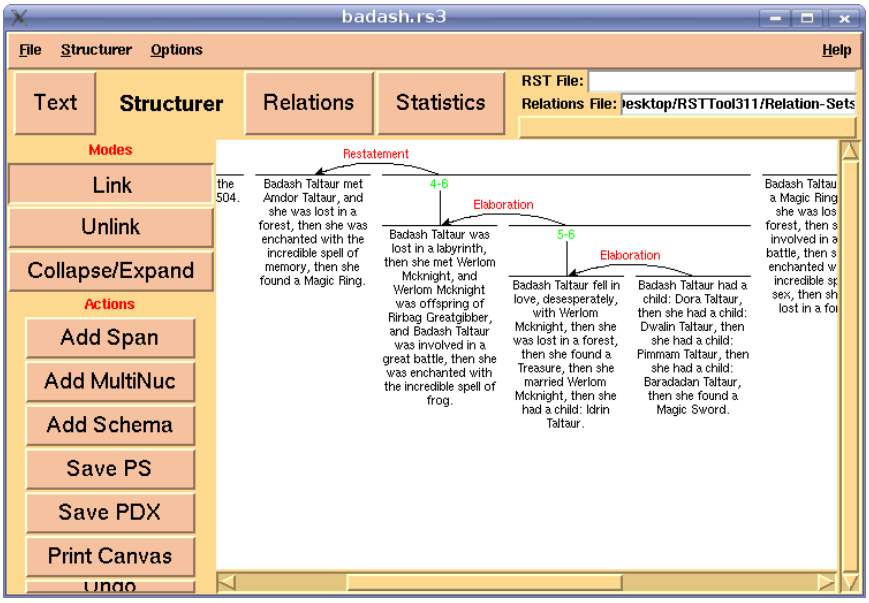

Figura 2.2: Captura de la aplicación RSTTool.

tiene una aplicación muy directa a la GLN [TM05]. En esta sección vamos a dar una introducción a las más importantes o que más influencia han ejercido en la investigación sobre la generación de textos o historias, o sobre las misma RST.

ILEX [OM98, OMOK01] es un sistema de generación de lenguaje natural desarrollado en la Universidad de Edimburgo que puede generar texto a partir de una serie de especificaciones del usuario como entradas del sistema. Estas especificaciones son el objetivo de la comunicación, la longitud total que se desea, y el perfil del usuario. Puede generar textos en español e inglés, en diferentes formatos (texto plano, HTML). Ilex es capaz de realizar todo el proceso de generación, desde la decisión sobre lo que se cuenta hasta la realización superficial en la que se genera texto. Esta aplicación es capaz de generar descripciones de un conjunto de objetos de una base de datos. Comenzó como un programa orientado a describir piezas de un museo, pero su funcionalidad de ha extendido de modo que puede realizar descripciones más complejas.

En [Hov93] se presenta un sistema que consiste en un interfaz para una base de datos que almacena datos sobre submarinos, y con el sistema de generación de lenguaje natural consigue extraer esta información. Esta tarea requería convertir relaciones retóricas en planes de discurso. Este tipo de aplicaciones, son muy usuales en la GLN.

Crear resúmenes de textos es otra utilidad grande de la RST. Suele basarse, como se propone en [Mar97a, Mar00, SJ95], en estudiar dónde están los núcleos de las operaciones y eliminar los satélites que les correspondan,

creando nuevas estructuras retóricas en las que sólo los núcleos que aparecían en las originales estén descritos, obteniendo así un resumen del texto original. Existen muchos más ejemplos de la creación de resúmenes con esta técnica, como puede comprobarse en [RS96, OSM94, TM02].

Encontramos en [GHST97] que las relaciones retóricas no sólo son aplicadas a la creación de discursos narrativos, sino que también es posible aplicarlas a variaciones de entonación en un discurso hablado, usando programas de síntesis de habla. Además, [Mar97c] aplica la RST en el "sentido contrario", usándola para interpretar textos escritos en vez de para generarlos.

En [PZ03] vemos que se usa la RST con el objetivo de extraer significado de los textos. En este trabajo se estudia cómo el significado de las palabras cambia según su la posición que les haya sido asignada en la jerarquía de la estructura retórica de un texto. Relacionado con este trabajo tenemos también el examen de la coherencia de un texto [BBHC$^+$01, BKW$^+$98, BM03]. Como la RST estudia la coherencia de los textos, puede ser aplicada para comprobarla.

La RST influye también en la etapa de generación de lenguaje natural de planificación de las frases, según los trabajos de [Fox87, CMO02, CIMT00, Tet05], ya que en ellos se expone que la posición y la relación de las diferentes partes del discurso en relación con las demás ha de influir en la correcta generación de expresiones de referencia. Por ejemplo, el uso de un pronombre para referirse al sujeto de una oración dependerá de la posición retórica de la oración en ese texto, y la de las demás que tengan relación con ella.

Como vemos, existe una amplia variedad de posibles aplicaciones de la Teoría de la Estructura Retórica, y de diversos tipos. Esto es un indicativo de su calidad, ya que desde 1985 lleva estudiándose y revisándose, no sólo por parte de los autores originales, sino de buena parte de la comunidad científica dedicada a la lingüística.

2.2. Generación de Historias en Lenguaje Natural

Atendiendo a las ideas presentes en [CL01], habitualmente queda separada la generación pura de textos en lenguaje natural de la creación narrativa de historias. Por una parte, la generación de lenguaje natural se centra en los asuntos lingüísticos en el ámbito de las frases, mientras que la generación narrativa pretende resolver el problema de hilos de historia y personajes, refinando, generalmente, sucesivamente la solución desde los objetivos globales de la narración hasta las acciones de los personajes, progresivamente dando más detalle. Por este motivo dedicamos una sección aparte a la Generación de Historias en Lenguaje Natural.

En este apartado de la investigación existen una gran cantidad de propuestas de muy diferentes características. Vamos a exponer un conjunto que

de manera suficiente apoye el trabajo realizado durante esta investigación.

2.2.1. Vladimir Propp

En relación con este trabajo tiene sentido mencionar a Vladimir Propp, formalista ruso que creó un análisis de los cuentos de hadas de la cultura rusa, y analizó sus principales características, que él consideró como atómicas. Construyó una topología con estos elementos de los cuentos que había descubierto, realizando, por tanto, uno de los primeros estudios formales de la narración de historias [Pro68]. Identificó, entre otros, FUNCIONES dentro de una historias y PERSONAJES que las realizaban. Las FUNCIONES son las de la enumeración de la enumeración siguiente [wik07]:

1. **Alejamiento**. Uno de los miembros de la familia se aleja.
2. **Prohibición**. Recae una prohibición sobre el héroe.
3. **Transgresión**. La prohibición es transgredida.
4. **Conocimiento**. El antagonista entra en contacto con el héroe.
5. **Información**. El antagonista recibe información sobre la víctima.
6. **Engaño**. El antagonista engaña al héroe para apoderarse de él o de sus bienes.
7. **Complicidad**. La víctima es engañada y ayuda así a su agresor a su pesar.
8. **Fechoría**. El antagonista causa algún perjuicio a uno de los miembros de la familia.
9. **Mediación**. La fechoría es hecha pública, se le formula al héroe una petición u orden, se le permite o se le obliga a marchar.
10. **Aceptación**. El héroe decide partir.
11. **Partida**. El héroe se marcha.
12. **Prueba**. El donante somete al héroe a una prueba que le prepara para la recepción de una ayuda mágica.
13. **Reacción** del héroe. El héroe supera o falla la prueba.
14. **Regalo**. El héroe recibe un objeto mágico.
15. **Viaje**. El héroe es conducido a otro reino, donde se halla el objeto de su búsqueda.
16. **Lucha**. El héroe y su antagonista se enfrentan en combate directo.

17. **Marca**. El héroe queda marcado.
18. **Victoria**. El héroe derrota al antagonista.
19. **Enmienda**. La fechoría inicial es reparada.
20. **Regreso**. El héroe vuelve a casa.
21. **Persecución**. El héroe es perseguido.
22. **Socorro**. El héroe es auxiliado.
23. **Regreso de incógnito**. El héroe regresa, a su casa o a otro reino, sin ser reconocido.
24. **Fingimiento**. Un falso héroe reivindica los logros que no le corresponden.
25. **Tarea difícil**. Se propone al héroe una difícil misión.
26. **Cumplimiento**. El héroe lleva a cabo la difícil misión.
27. **Reconocimiento**. El héroe es reconocido
28. **Desenmascaramiento**. El falso queda en evidencia.
29. **Transfiguración**. El héroe recibe una nueva apariencia.
30. **Castigo**. El antagonista es castigado.
31. **Boda**. El héroe se casa y asciende al trono.

Y los PERSONAJES que identificó Propp, éstos. Hay que tener en cuenta que no representa exactamente los personajes que se indican, sino más bien los *roles* que desempeñan. Por ejemplo, la princesa puede ser un tesoro:

1. El **villano**, que está en contra del héroe.
2. El **donante**, que prepara al héroe o le da un objeto mágico.
3. El **ayudante**, que ayuda al héroe en la búsqueda.
4. La **princesa**, con quien el héroe se casa.
5. El **padre de la princesa**, que no siempre está diferenciado de la princesa.
6. El **despachador**, que hace que se conozca la parte oculta de la historia, y envía al héroe a la búsqueda.
7. El **héroe**, que aprende del donante, y se casa con la princesa.
8. El **antihéroe**, que intenta usurpar la posición del héroe procurando casarse con la princesa.

2.2.2. Aplicaciones concretas de generación de historias en lenguaje natural

En BRUTUS [BF99] tenemos un sistema que pretende ir más allá de la creación de narraciones de historias, sino que intenta crear versiones distintas de una misma historia. Es interesante esta propuesta porque afronta el problema de no sólo crear una historia desde cero, sino también recibirla sin conocimiento de la creación previa, como los sistemas de modificación de historias. BRUTUS maneja una cantidad muy grande de información, y se aspira a que genere textos de calidad parecida a la humana.

AUTOMATIC NOVEL WRITER [KAB+73], uno de los sistemas más antiguos que se estudian aquí, recibe la entrada del usuario de modo que se establecen los personajes y demás datos de la historia, y crea narraciones en un inglés algo pobre.

TALE-SPIN, un clásico sistema de generación de historias, incorporando ideas de la Teoría de la Dependencia Conceptual de Roger Schank [Sch69]. En este sistema un conjunto de personajes son "encerrados" en un mundo determinado con unas reglas, y se les asigna unas tareas que tienen que cumplir, posiblemente ayudados por el usuario. No existe la posibilidad de que los personajes tomen decisiones creativas, ya que todo el sistema está fundamentado en una planificación algo rígida. El problema principal de TALE-SPIN es que las historias que surgen pueden no tener el más mínimo interés, y carecer de coherencia.

STORYBOOK [CL02] se ofrece como toda una arquitectura de generación de historias, con unos resultados buenos, aunque muy ceñidos a los ejemplos concretos de estudio (Caperucita Roja). Consigue una calidad literaria en todas las operaciones de la generación. Su generador de prosa, AUTHOR, mezcla las gramáticas de producción con técnicas de agrupación, más clásicas en la GLN.

MINSTREL [Tur92] enfoca la generación de historias como réplica de la manera que un trovador medieval tiene de contar las historias de gesta. El sistema almacena los episodios acontecidos como un cuentacuentos, en una base de conocimiento. El usuario propone una moraleja, y el sistema procura, a través del examen de una serie de conflictos que surgen entre los distintos personajes que componen la historia, crear un discurso coherente. MINSTREL se centra en el dominio de las historias legendarias de la leyenda artúrica.

MEXICA [PyPS01] es una aplicación que se engloba dentro de la generación creativa de historias que genera "esqueletos", siguiendo dos pasos principales. Uno es la *fase de compromiso*, que, siguiendo restricciones y estructuras retóricas, crea una historias. Después, la *fase de reflexión* evalúa la novedad de la historia generada y verifica que, realmente, existe coherencia. MEXICA une técnicas de resolución de problemas objetivos que conducen la generación de historias. MEXICA es un trabajo directamente comparado al

que proponemos aquí, ya que se basa en la planificación de contenido. La diferencia principal en cuanto a salida del sistema es que MEXICA no genera diálogos ni realiza las operaciones de creación final del texto.

El trabajo sobre el que podemos leer en [RY06] parte de un estado inicial del mundo e intenta generar historias siguiendo un conjunto de objetivos, muy en la línea de nuestra propuesta, como veremos. De la misma manera que el sistema anterior (MEXICA), no está orientado a la generación de textos finales, (sólo realiza tareas de planificación de contenido), y no considera los diálogos como una parte de la planificación de la narración.

Aparte de estas aportaciones, tenemos las que podemos encontrar en el trabajo de David E. Rumelhart [Rum75], que usa gramáticas para generar historias, STARSHIP [Deh89], que lega la planificación al autor pero le guía en la conclusión de los objetivos, o HOMER [Kon01], que se orienta a la producción multimodal (gráficos y texto).

2.3. Creatividad computacional

La primera pregunta susceptible que surge al pretender definir la CREATIVIDAD COMPUTACIONAL es la de definir, previamente, qué es la *creatividad*. La definición de la REAL ACADEMIA ESPAÑOLA (RAE) la podemos ver en la Figura 2.3.

creatividad.

1. f. Facultad de crear.
2. f. Capacidad de creación.

Figura 2.3: Definición de *creatividad* ©Real Academia Española.

Sin embargo, esta definición no capta la semántica que usualmente se le da a la creatividad. Cuando decimos de algo o alguien que es creativo, generalmente nos estamos refiriendo a una propiedad o capacidad de resolver problemas de una manera original, inesperada o poco común, de una manera que, para la mayoría de los individuos no es usual [Wig06]. Sin embargo, cualquier definición que se le pueda dar a la creatividad, tal y como acabamos de hacer, que pretenda resumir el significado que suele atribuírsele a la palabra se usan metáforas y términos con poco significado concreto, de modo que la definición de lo que solemos entender por *creatividad* no es sencilla de establecer de un modo formal que nos permitiera discernir qué es creativo y qué no, entendiendo no la definición de la RAE.

Por tanto, es tentador terminar el estudio del estado de arte de la CREATIVIDAD COMPUTACIONAL concluyendo que la creatividad sólo es otro

término del lenguaje del que no tiene sentido hablar ni hacer un estudio, siguiente las ideas de Wittgestein [Wit22]. Y, aunque desde el punto de vista de la Filosofía esto puede ser cierto, desde el enfoque de la Informática y la Inteligencia Artificial, como emuladoras del ser humano, sí tiene interés estudiar qué es lo que hace una obra *creativa*, de modo que podamos, con los recursos de los que disponemos, crear máquinas capaces de comportarse como humanos creativos.

Podemos, siguiendo las ideas de [Wig06] decir que la CREATIVIDAD COMPUTACIONAL es:

> "El estudio y el apoyo, a través de intenciones y métodos creativos, del comportamiento exhibido por sistemas naturales y artificiales que serían considerados como creativos si fueran exhibidos por humanos".

A pesar de que esta definición pueda parecer apoyada en sí misma o "circular", no es así. Lo que se intenta es capturar conceptualmente una serie de ideas que son difíciles de definir intensionalmente, pero que existen en el mundo humano.

Existen varios trabajos relacionados que no solamente exploran las posibilidades de la creatividad desde un punto de vista descriptivo, sino que intentan ofrecer soluciones a problemas clásicos de la Inteligencia Artificial en General y de la Generación de Lenguaje Natural en particular, como, por ejemplo, [Vea06], que estudia las posibilidades de expresión de las analogías, [Rit06], que estudia la creatividad desde el punto de vista transformacional, intentando encontrar requisitos o características que la confieran o que pongan de manifiesto unas propiedades más empíricas para que el estudio sobre la creatividad computacional pueda llevarse a cabo de manera más sencilla.

También, desde otros puntos de vista, tenemos la evaluación de la creatividad [PG06], o la aplicación de técnicas de creatividad con los ordenadores para el reconocimiento de formas [OBK06].

2.3.1. Evaluación de la calidad de las historias

Evaluar un texto generado con una máquina dista mucho de ser una tarea sencilla. Surgen una cantidad de problemas grande cuando se procura discernir cuándo un resultado es buena o mala calidad. Sólo hemos de darnos cuenta de que no sólo en la evaluación del resultado de la máquina aparece este inconveniente, sino en la misma producción literaria humana surgen, muy a menudo, problemas.

Además, evaluar un texto no sólo es cuestión de dar una nota, ya que un texto está definido por varias características como la coherencia, la legibilidad, el interés, etcétera, que pueden (y en ocasiones es necesario) ser evaluadas por separado, y es algo que hay que tener en cuenta. No obstante,

en lo que a nuestra investigación respecta, la evaluación que vamos a observar es la que está relacionada con la creatividad y el parecido de los textos a lo que los humanos generarían, y este límite es sobre el que presentamos las siguientes ideas.

Desde el punto de vista ingenieril de la mejora de los productos con base científica, es necesario un sistema de evaluación de la producción que nos ayude. Se han diseñado métodos de diferentes tipos que ponen a nuestro alcance algunos métodos de valoración.

Por un lado, tenemos la comparación directa con textos del mismo contenido que han sido generados por humanos, que sólo es, evidentemente, posible, si se dispone de un *corpus* de textos con los que cotejar el resultado. Esta aproximación no suele ser posible, dada la naturaleza de los textos generados. Por otro, tenemos la posibilidad de realizar una encuesta a expertos (o no expertos) sobre el material generado que puedan dar una valoración al producto con la que retocar la generación de las historias, como en [PG06].

2.4. XML

El lenguaje extendido de marcado XML [Har01] se está convirtiendo, cada vez más, en el estándar de facto para el marcado de todo tipo de documentos estructurados. XML está basado en SGML, del que es una versión más simple y orientada al uso. Una parte muy importante del uso de XML, hoy en día, se centra en la aplicación de estructura mediante el marcado de documentos en los que los terminales del lenguaje (el contenido en sí del documento), son, en su mayor parte, y junto con los atributos de las etiquetas, valores correspondientes a estructuras de datos representadas, mediante XML, en un formato de texto plano.

XML marca los documentos mediante *etiquetas*, que son símbolos que encierran, de una manera anidada, los contenidos de un conjunto de datos. Estas etiquetas pueden tener atributos, que ajustan su semántica o dan más información al marcado de cierto símbolo terminal del lenguaje que se marca. Podemos ver un ejemplo de un archivo XML en la figura 2.4.

Un documento XML, en lo relativo a su corrección, tiene dos propiedades: la de ser *bien formado* y *válido*. Un documento XML está bien formado cuando su sintaxis sigue las normas generales de XML y su estructura es correcta (por ejemplo, con una anidación apropiada). Sin embargo, para que un documento sea válido es necesario que cumpla una serie de normas definidas en una gramática concreta. Este tipo de gramáticas puede ser especificada en documentos de varios tipos.

2.4.1. DTD

Las Definiciones de Tipo de Documento o *Document Type Definitions* (DTD) son documentos que tienen una gramática diferente a las de los

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE agenda SYSTEM "agenda.dtd">

<agenda propietario = "Juan Montes">
  <registro> Luis Toledo </registro>
  <registro> Marta Prado </registro>
</agenda>
```

Figura 2.4: Ejemplo de XML.

documentos XML, pero que definen de una manera legible y sencilla de especificar cuáles son las reglas que gobiernan la validez de estos documentos [dtd07]. Mediante una DTD concreta es posible establecer qué documentos XML son válidos para un dominio y un uso determinados, y cuáles no.

La ventaja principal de las DTDs respecto de otras posibles alternativas es su sencillez. La sintaxis con la que se representa una gramática válida es muy simple. Tienen como desventaja que no son la recomendación del W3C, y que su sintaxis no es XML. En la Figura 2.5 podemos ver un ejemplo de DTD posible para el archivo XML de la Figura 2.4.

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT agenda (registro*)>
<!ATTLIST propietario id CDATA #REQUIRED>

<!ELEMENT registro (#PCDATA)>
```

Figura 2.5: Ejemplo de DTD.

2.5. Ontologías

Para establecer el significado de los datos es posible añadir información a los repositorios de datos disponibles, de modo que tengamos un conjunto de etiquetas o marcas sobre los datos que los ordenadores puedan procesar. Hay que tener en cuenta, sin embargo, que una máquina no "comprende" el marcado de la información. Para que el marcado tenga un uso real, hace falta además un consenso en cuál ha de ser la interpretación de las marcas, o unidades que añaden información para el proceso, con las que anotamos los datos cuyo significado una máquina no puede inferir. Las ontologías pueden ser usadas para especificar esta semántica.

Una ontología es un sistema de definición exacta de entidades y las re-

laciones entre ellas. Una ontología puede definirse como un "artefacto ingenieril" que:

- Está constituido por un vocabulario específico que representa una realidad determinada, y
- Un conjunto de asunciones explícitas que establecen el significado de los símbolos de ese vocabulario.

De este modo, las ontologías ofrecen un mecanismo para compartir información de un dominio de interés concreto y una especificación formal para que las máquinas puedan interpretar y manipular información de ese dominio. Por otro lado, las ontologías no son únicamente un recurso tecnológico relacionado con la informática. Una ontología define un vocabulario de una manera formal que puede ser usado por diferentes entidades (personas, empresas, etc.) de una manera unívoca.

Según [NM01], los motivos principales para desarrollar una ontología son los siguientes:

- En general, siguiendo la opinión de [Mus92, Gru93], podemos decir que es una de las principales metas para desarrollar ontologías.
- tenemos la posibilidad de reusar las ontologías ya creadas, e incluso extenderlas para crear nuevo conocimiento en forma de ontología.
- Las reglas gobiernan la lógica de los sistemas de cierto dominio, usualmente, ha venido siendo especificada en un lenguaje de programación. Es complicado, por tanto, el descubrimiento y cambio, a posteriori, de estas reglas. Las ontologías permiten la , de modo que en el mismo conocimiento del mismo podemos manejar sus reglas de comportamiento como sistema.
- es otra de las principales ventajas de las ontologías, ya que queda desacoplado el conocimiento del uso que se hace de él.
- Analizar el conocimiento de un dominio. En [MFRW00] se expone que el análisis formal de un bloque de conocimiento almacenado como una ontología es muy valioso para reusarlas y extenderlas.

En los últimos años el uso de las ontologías ha ido extendiéndose desde sus inicios en los estudios en el campo de la Inteligencia Artificial hasta las especificaciones de expertos en dominios muy distintos, que desarrollan ontologías estandarizadas para compartir y anotar información en sus entornos de trabajo. Las ontologías están siendo muy utilizadas en la *web*. Estas ontologías van desde grandes taxonomías que clasificacan la información (como

en YAHOO![3]) a categorizaciones y descripciones de productos para venderlos (como en AMAZON[4]). En el ámbito de otras disciplinas también se usan ontologías

Para describir ontologías en este trabajo se ha optado por usar OWL, como se explica más adelante. A continuación exponemos una breve introducción a RDF, por ser la tecnología sobre la que está construido OWL, y, después, a OWL propiamente dicho.

2.5.1. Resource Description Framework (RDF)

El FRAMEWORK DE DESCRIPCIÓN DE RECURSOS (RDF, por sus siglas en inglés) es un lenguaje de propósito general orientado a la representación de información en la *web*. Su uso se centra en el desarrollo de la *web semántica*, y tiene como finalidad describir los recursos de la misma de una manera no orientada a la legibilidad por parte de un humano, sino a la computación de la información contenido por un ordenador.

La *web semántica* es un proyecto de futuro en el que la información web tiene un significado exactamente definido y puede ser procesado por ordenadores. Por lo tanto, los ordenadores pueden integrar y usar la información disponible en la web. Más información sobre la *web semántica* puede ser encontrada en [sem07]

RDF está considerado como un lenguaje de *metadatos*, o "datos sobre datos", ya que con los símbolos de RDF añadimos *metainformación* a los datos que realmente nos interesan para poder interpretarlos de una manera exacta, es decir, de aquella que el autor de los datos (o, al menos, del autor del marcado RDF sobre estos datos) quería que estos fuesen interpretados.

RDF define los recursos mediante descripciones de los mismos, como puede verse en el listado 2.6. Puede encontrarse más información sobre la manera de la tecnología RDF de describir estos recursos en [rdf07].

```
<rdf:Description
 rdf:about="http://www.recshop.fake/cd/Empire Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>
```

Figura 2.6: Ejemplo de un recurso RDF.

RDF no define clases de datos específicas para las aplicaciones. En vez de esto, el estándar RDF dispone de RDF SCHEMA. Estos documentos

[3]http://www.yahoo.com
[4]http://www.amazon.com

definen nuevas clases, y relaciones entre ellas (herencia, agregación) de una manera muy similar a aquella con la que se acostumbra en la programación orientada a objetos. En la Figura 2.7 podemos ver un ejemplo de un RDF Schema.

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf= "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base=  "http://www.animals.fake/animals#">

<rdf:Description rdf:ID="animal">
  <rdf:type
   rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
</rdf:Description>

<rdf:Description rdf:ID="horse">
  <rdf:type
   rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="#animal"/>
</rdf:Description>

</rdf:RDF>
```

Figura 2.7: Ejemplo de un RDF Schema.

2.5.2. Web Ontology Language (OWL)

De la misma manera que RDF, OWL nace de la Web Semántica. Se convirtió en una recomendación del W3C (World Wide Web Consortium) en febrero del año 2004. Respecto de otros sistemas de representación de conocimiento, OWL presenta ventajas para grandes sistemas en los que hay una gran cantidad de información relevante almacenada, y es necesario un tráfico de la misma que permita interconexión entre las partes de la comunicación de una manera sencilla y utilizable desde muy diversas plataformas. OWL está basado en RDF, y, por tanto, en XML. Permite que las máquinas representen, entiendan y procesen de un modo más cómodo la información sobre las ontologías que está almacenada computacionalmente. Esto coloca esta tecnología con una ventaja sobre las anteriores.

Respecto de estas tecnologías, OWL añade más vocabulario para describir propiedades y clases: entre otras, relaciones entre clases (por ejemplo, de elementos "disjuntos"), cardinalidad ("exactamente uno"), igualdad, propiedades con un conjunto de tipos más rico, características de las propiedades ("simetría") y clases enumeradas.

Sublenguajes de OWL

La sintaxis de OWL permite expresar tres dialectos distintos de este lenguaje. Cada uno de ellos permite representar las mismas taxonomías de conceptos, pero la capacidad de expresar las relaciones como propiedades entre los mismos varía. Estos tres sublenguajes son OWL Lite, OWL DL y OWL Full. Sus capacidades expresivas van, por ese orden, en aumento. En la siguiente lista damos más información sobre estas tres posibilidades de crear ontologías con OWL.

- **OWL Lite** permite realizar una clasificación de conceptos y especificar restricciones simples. Por ejemplo, aunque da soporte a restricciones de cardinalidad, sólo permite valores de cardinalidad de 0 o 1. Debería ser más simple proveer soporte mediante herramientas para este dialecto, y, además, OWL Lite permite una migración simple y casi directa en muchos casos a partir de Tesauros y Taxonomías. Además, desde luego, es mucho menos complejo que sus "hermanos mayores".

- **OWL DL** da soporte a aquellos usuarios que necesitas mucha más expresividad para la ontologías mientras que también se mantiene la completitud computacional (se garantiza que todas las conclusiones son computables) y la decidibilidad (se sabe que todas las computaciones acaban en un tiempo finito). Este dialecto incluye todas las construcciones de OWL, pero sólo pueden ser usadas bajo ciertas restricciones (por ejemplo, aunque una clase puede ser subclase de muchas clases, una clase no puede ser una instancia de otra clase). El nombre que este dialecto recibe se debe a su correspondencia con las Lógicas Descriptivas.

- **OWL Full** se ha diseñado para aquellos usuarios que necesitan un control total sobre las ontologías, permitiendo una expresividad de máximo nivel, y la libertad sintáctica que provee RDF sin garantías sobre la computación como tenía OWL DL. Por ejemplo, en este sublenguaje una clase puede ser tratada de manera simultánea como una colección de individuales y como un individual propiamente dicho. Permite que una ontología aumente el significado predefinido de RDF u OWL. Con toda esta complejidad, no existe ningún software capaz de controlar el 100 % de los aspectos de este sublenguaje.

Cada uno de estos sublenguajes es un subconjunto del siguiente. Es decir, las ontologías que pueden describir son subconjuntos de aquellas que pueden describir los lenguajes más potentes:

$$\text{OWL Lite}_{ontologia} \subset \text{OWL DL}_{ontologia} \subset \text{OWL Full}_{ontologia} \tag{2.3}$$

Del mismo modo, las conclusiones que pueden inferirse, siguen el mismo patrón:

$$\text{OWL LITE}_{conclusion} \subset \text{OWL DL}_{conclusion} \subset \text{OWL FULL}_{conclusion} \quad (2.4)$$

Capítulo 3

Sistema de generación de historias

Como se ha avanzado hasta aquí, el objetivo de este trabajo es crear un sistema capaz de narrar sucesos que ocurren entre personajes como una historia textual, basándonos en partes narradas y diálogos. Partiendo del estado del arte que se ha expuesto en el capítulo anterior, se ha diseñado e implementado un sistema capaz llevar a cabo esta tarea que ofrece algunas aportaciones a los trabajos antes comentados. Entre ellas se encuentran la capacidad de narrar historias de varios personajes en una sola historia, y de crear historias con partes de narración y partes de diálogos entre los personajes.

El sistema de generación de narraciones requiere datos de entrada para disponer de conocimiento a partir del cual generar las historias. Este conocimiento viene dado de un conjunto de aplicaciones, algunas de las cuales se presentan en este trabajo. Este conjunto puede, por supuesto, ser ampliado con más aplicaciones siempre y cuando éstas tengan como salida datos que cumplan las especificaciones de entrada a nuestro sistema de generación de historias.

Estas aplicaciones, que son presentadas y discutidas en la Sección 4.2, tienen el denominador común de ser programas en los que se desarrollan una serie de operaciones, vidas o trazas de ejecución de varios personajes o agentes (según el caso) de forma que cada entidad interacciona con las demás en mayor o menor medida, muchas veces con diálogos de diversa índole: desde la transmisión textual de información de alto nivel hasta la comunicación de robots autónomos para realizar una operación de campo. Veremos más adelante detalles sobre estos sistemas.

Al operar con sistemas con varios personajes que desarrollan trazas de ejecución entre sí, tenemos, pues, un sistema en el que la narración textual y el diálogo son elementos que se complementan perfectamente para llevar a cabo la tarea de narrar lo sucedido. Por este motivo, como ejemplo del

sistema se presenta la narración de un diálogo de Platón, el *Menón* [Pla]. Se ha creado una descripción conceptual del mismo y se ha ampliado con elementos narrativos, para crear una versión generada por la máquina de éste diálogo. Se pretende que este ejemplo sirva como sustrato sobre el que ejemplificar de manera clara el funcionamiento, en algunas partes complejo, del sistema.

Las razones de haber escogido uno de los diálogos de Platón para ejemplificar todo el sistema que se ha desarrollado son las siguientes:

- Cada diálogo de Platón tiene un mensaje general claro que quiere transmitir. Esta característica habilita la posibilidad de partir de un solo objetivo claro y común a partir del cual ir dividiendo subobjetivos sucesivamente.

- Los diálogos de Platón son una fuente de datos, como *corpus*, de las que podemos extraer diálogos e inferir, a partir de los textos que tenemos escritos, partes narrativas con las que completar éstos y crear así una historia híbrida con narraciones y partes habladas.

- Disponemos de un *corpus* lo suficientemente extenso como para poder extraer toda la información que necesitamos dentro de un mismo dominio.

A pesar de que, por supuesto, estas características existen en una infinidad de otros textos, los diálogos de Platón son un ejemplo sencillo de las mismas, y perfectamente accesible.

El esquema de la funcionalidad del sistema de forma global está representado en la Figura 3.1. Los bloques cuadrados representan las tres etapas de proceso que se efectúan en los datos: la *importación y traducción de la entrada*, para lo cual usamos una ontología; la *generación del discurso*, en la que utilizamos, por un lado, el objetivo del usuario, o aquello que desea que se cuente, y por otro reglas de producción para la generación; y por último la *creación del texto*, en la que damos a los datos ordenados la forma de texto legible. Tal como se presenta en la figura, es necesario dar unas reglas de generación que incluyan en el sistema la información sobre el idioma en el que se genera el texto.

El capítulo está dividido en cuatro secciones principales, en las que se explican con detalle, y en el orden lógico secuencial de generación de un texto, las partes principales del proceso. Primero se explica cuáles son las características que han de tener los datos de entrada en el sistema de generación, y cómo los hemos obtenido en la implementación del trabajo de investigación, en la Sección 3.1. Después, en la Sección 3.2 mostramos el núcleo del sistema generador, cómo se procesan los datos de entrada para generar el contenido conceptual de la historia. La sección siguiente (Sección

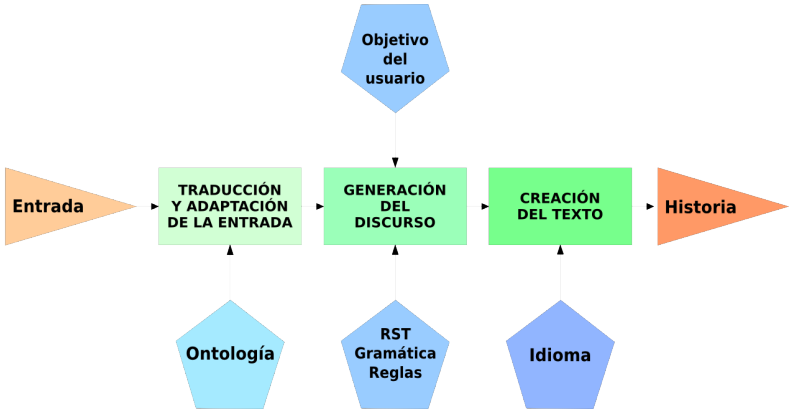

Figura 3.1: Diagrama que representa el funcionamiento de tubería de la aplicación de generación de historias.

3.3) explica el módulo de realización textual del generador, que se encarga de traducir la representación estructurada y poco legible del módulo de generación de discurso a un lenguaje natural. En la última sección (3.4) mostramos un ejemplo completo del funcionamiento del sistema.

3.1. Entrada de datos al sistema

El sistema que presentamos necesita ser alimentado con datos que forman, junto con el conocimiento del dominio, el bloque de información que tiene de entrada el algoritmo de generación. Desde luego, una vez establecido un estándar de entrada, cualquier aplicación que contenga la información requerida puede ser una fuente de datos válida para nuestro programa. El el Capítulo 4.2 se presentan algunas aplicaciones que han sido usadas para el prototipado del sistema, y, como se podrá comprobar, son fuentes muy diversas.

A continuación mostramos qué datos necesita como entrada el programa, los diferentes formatos que se han utilizado para alimentar a la aplicación de generación de historias, y cuáles han sido las motivaciones de su uso y creación.

3.1.1. Ontología del dominio

Como se ha expuesto en la Sección 2.5, las ontologías permiten almacenar conocimiento de un dominio de una manera independiente de la aplicación. Esto ofrece un gran número de ventajas: modularidad, reusabilidad y encapsulación, entre otras. Las ontologías, pues, nos ofrecen una solución al problema de la importación desde diferentes dominios que acabamos de comentar, permitiéndonos definir un conocimiento que está fuera de la

aplicación, y también del conjunto de datos particular sobre el que se va a trabajar, ya que somos capaces, usando ontologías, de definir el modelo lógico del sistema de una manera independiente al puro programa de ordenador, y de este modo las otras aplicaciones que generen datos que, a través del programa que proponemos, vayan a ser traducidos a una historia, pueden conocer el modelo.

Hoy en día no es posible dar una descripción general del conocimiento que el ser humano usa para generar historias. Por eso, hemos de ceñirnos a una descripción dependiente del dominio, mucho más particular y sencilla de definir. Para esto, se ha creado una ontología en la que se almacenan los diferentes elementos que componen nuestro dominio sobre las historias y las relaciones entre ellas. La ontología que se ha usado tiene una estructura simple, y aprovecha, principalmente, las características de taxonomía que tienen las ontologías. No se usa una gran cantidad de reglas ni propiedades, ya que en el prototipo que se presenta no ha sido necesaria una descripción exhaustiva de los contenidos formales del modelo lógico del sistema.

Hay que notar que no sólo el modelo del sistema, como taxonomía de conceptos, es necesario para que se puedan crear datos para la narración. Para ser capaz de crear un contenido coherente, la aplicación requiere tener también un conjunto de datos que ha de ser capaz de procesar según unos algoritmos determinados, como explicamos en la Sección 3.2. Además, en estos datos hemos de incluir en los datos una información sobre cada uno de ellos que nos permita decidir, en la etapa de proceso, cómo han de ser tratados. La propuesta sobre la representación de la información a partir de la que crear la historia se ha basado en muchas alternativas clásicas previas, como se puede encontrar en [RD00].

Descripción de la ontología

La división que se ha hecho de los datos tiene un raíz global. La razón de disponer de un tipo común de elementos de la historia para el proceso es el hecho de que, en los eventos que pueden darse, cualquier cosa puede ser referenciada, no sólo entidades físicas. Por ejemplo, en la frase:

Sócrates enseña Filosfía.

Tenemos, evidentemente, que tanto “Sócrates” como “Filosofía” son referentes, es decir, podemos hacer referencia a ellos en una frase cualquiera del texto. Sin embargo, hay que tener en cuenta que en frases como:

Sócrates le dijo a Menón que la virtud podía ser conocida.

Vemos que no sólo “Sócrates” es un referente, sino que también la oración que dice “la virtud podía ser conocida” ha de poder ser referenciada, como objeto directo de la oración. Por tanto, consideramos que todo puede ser

un referente. De esta manera la concatenación del discurso basada en las relaciones entre los referentes, que es como se hace en este trabajo (y se explica en la Sección 3.2) puede ser mucho más rica, ya que en cada sentencia tenemos varias relaciones a varios referentes de todos los tipos posibles.

Como es necesario que nuestro sistema tenga diálogos, el hecho de que todas las entidades de la historia sean REFERENTES nos permite que en las partes habladas los personajes puedan hacer referencia a cualquier parte del conocimiento de la historia. Si vemos, por ejemplo, el siguiente diálogo:

> *Sócrates dijo: Yo conozco la virtud.*
>
> *Menón dijo: Quiero que me enseñes la virtud porque tú conoces la virtud.*

La representación interna ha sido la de la Figura 3.2. En esta Figura y en el texto vemos que hay una relación de causalidad entre la intención de Menón y el hecho de que Sócrates conozca la virtud. Al ser los EVENTOS también REFERENTES, podemos hacer relación al EVENTO de que "Sócrates conozca la virtud", mediante la relación de CAUSALIDAD. En la Figura 3.3 anticipamos también cómo se especificaría esta relación en un formato que se ha desarrollado para el sistema.

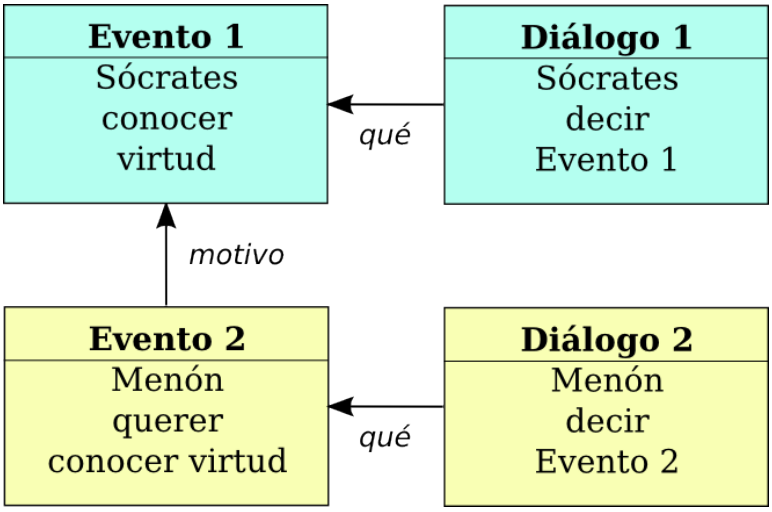

Figura 3.2: Representación gráfica del discurso.

En resumen, todos los datos con los que se trabaja heredan de un mismo concepto de la ontología, el REFERENTE. En la Figura 3.4 se muestra el árbol de herencia de los REFERENTES, según la taxonomía definida en la ontología de nuestro sistema.

Para explicar un poco mejor esta figura, vamos a detallar los conceptos que hay reflejados en la Figura 3.4 en la siguiente explicación de la taxonomía:

```
character socrates

character meno

event ev1 {
  who socrates;
  verb know;
  direct virtue;
}

event ev2 {
  who meno;
  verb want_to_know;
  direct virtue;
  because ev1;
}

dialogue di1 {
  who socrates;
  verb say;
  direct ev1;
}

dialogue di2 {
  who meno;
  verb say;
  direct ev2;
}
```

Figura 3.3: Representación del discurso en el lenguaje de especificación propio (ver Sección 3.1.2).

- **Referentes**. El generador de historias que se presenta está enfocado en la narración de sucesos acontecidos a personajes. Por lo tanto, la representación de los mismos que se cree es fundamental para el sistema.

  Hay que tener en cuenta, no obstante, que no sólo los personajes son los elementos de la historia. Si un personaje "rompe una piedra", la piedra también tiene entidad semántica, y es fundamental reconocerla como elemento en la historia para hilar la narración con posibilidades de relación entre hechos más amplias, ya que si olvidamos "la piedra", no sabremos que la "piedra que rompió" puede ser, por ejemplo, la misma que "la piedra que era de oro". Conocer la entidad piedra, es, por tanto, fundamental.

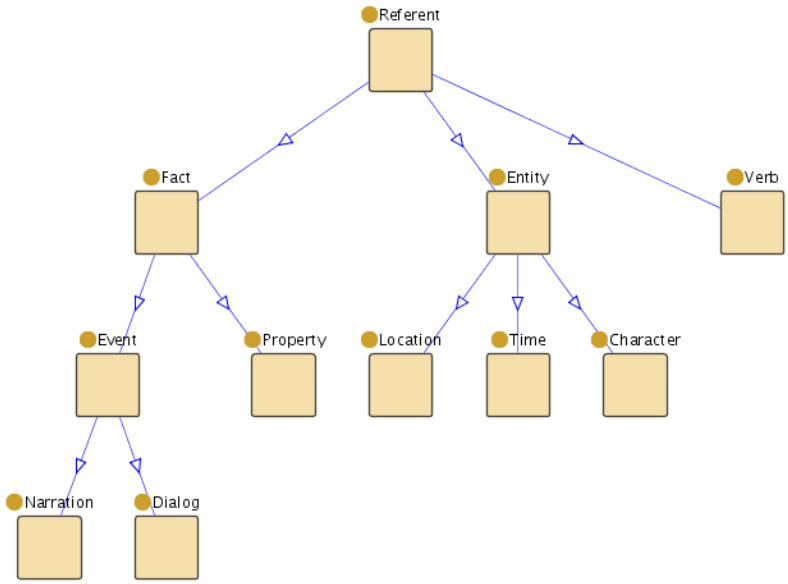

Figura 3.4: Árbol de la ontología de los referentes (en inglés, por su descripción en la ontología en el prototipo del sistema).

Así, dentro del sistema de generación, creamos el tipo REFERENTE, (que será un objeto de programación), y será la raíz de todos aquellos elementos semánticos y con entidad que puedan ser referidos durante el proceso de generación, ya sea como agentes, o elementos pasivos. De los referentes heredamos, con el sentido de la herencia en al programación orientada a objetos, tres tipos de objetos:

- **Entidades**. Las entidades, representan aquellos objetos que no tienen capacidad de actuar, y reciben acciones por parte de los personajes.
  - **Personajes**. Los personajes, evidentemente, representan los objetos con capacidad de actuar, y son correspondientes a aquellos que se reciben de los sistemas en los que se sucede su vida o ejecución.
  - **Lugares**. Los lugares son aquellas entidades que tiene un significado relacionado con un lugar. Estas entidades han de ser tratadas de forma especial, ya que en las narraciones las relaciones de lugar son fundamentales.
  - **Tiempo**. El tiempo es considerado en este modelo también como una entidad. De este modo podemos manejar sus refe-

rencias de una manera similar a cómo manejamos el resto de las entidades.

- **Hechos**. Las historias están formadas por conjuntos de hechos que se suceden durante el discurso narrativo. Sin embargo, los HECHOS no son simplemente elementos de información que podemos incluir o no en la historia en un punto determinado, sino realidades que acontecen en un tiempo y lugar. Pueden ser verdad durante toda la historia, en un instante, o un intervalo. Los hechos, como se ha comentado, pueden ser referenciados, y un personaje puede "contar" un hecho que exista en la historia. Los HECHOS pueden tener varias relaciones con REFERENTES. El conjunto de estas relaciones depende del tipo de HECHO.

  Se han divido los hechos en dos grupos: las PROPIEDADES y los EVENTOS:

  - **Propiedades**. Las propiedades cuentan alguna característica de un referente. Son el equivalente, en el lenguaje humano, a las frases copulativas, en las que no hay acción, sino descripción de propiedades. Por ejemplo, un hecho es: *Sócrates es mortal*.
  - **Eventos**. Los EVENTOS, a diferencia de las PROPIEDADES, relatan una acción. Son equivalentes a las frases predicativas en los lenguajes naturales. *Sócrates enseñaba Filosofía en Atenas* es un ejemplo de un EVENTO. Para este sistema en el que las partes habladas son tan fundamentales se han divido, además, los EVENTOS en dos subgrupos:
    - **Diálogo**. Necesitamos, para tratar la información de la que disponemos de un modo correcto, poder saber cuándo algo es "dicho" y cuándo no. Por eso, tenemos que los DIÁLOGOS son aquellos eventos en los que se dice algo, no sólo se efectúa.
    - **Narración**. Este tipo de eventos son los correspondientes a todas aquellas acciones que no son habladas. Corresponden a todas las acciones como "comer", "andar", "coger", etcétera.

- **Verbos**. Los verbos son las acciones, y, por lo tanto, el núcleo de los HECHOS. "Decir", "comer", "alistar" o "filosofar" son VERBOS válidos en nuestro sistema.

Además de esta taxonomía que acabamos de presentar para la ontología, se han definido también, aprovechando las posibilidades de las LÓGICAS DESCRIPTIVAS que ofrece OWL, una serie de propiedades entre los elementos de esta taxonomía de la ontología, de modo que incluyamos en el dominio

algo más de conocimiento, cosa que va a ser fundamental a la hora de crear las reglas que van a dirigir la generación de los textos.

Estas propiedades en el prototipo implementado son, por ejemplo, que el sujeto de un hecho ha de ser una ENTIDAD, o que los intervalos de acción de un evento tiene que ser TIEMPOS, y el orden que existe entre éstos. Sin pretender dar detalles más concretos de implementación explicando todas las propiedades, es importante notar que estas características son fundamentales para asegurar una correcta creación de la historia.

Instancias de referentes

Una vez que la ontología, y con ella la estructura del dominio del sistema y las relaciones que entre cada elemento del mismo existen, es posible crear INDIVIDUOS para el dominio, o aquellas entidades que realmente instancian las clases de la ontología: los elementos reales sobre los que van a ser creadas las historias. En la Figura 3.5 vemos esta idea representada gráficamente.

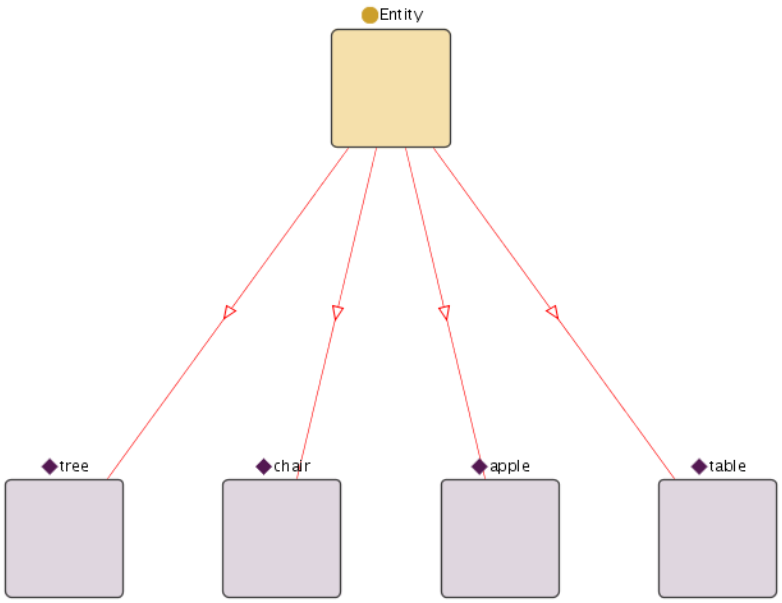

Figura 3.5: Ejemplo de individuos en el dominio de una historia.

Cada entidad de la historia pertenece a una clase de REFERENTE, ya sea un PERSONAJE o un LUGAR, por ejemplo. El único paso que resta, por tanto, es crear estas instancias. Existen diversas posibilidades para definirlas y crear así una descripción de la historia, como explicamos en la Sección 3.1.2. Sin embargo, no todo el conocimiento de cada historia tiene que estar

contenido en la descripción de la historia. Exponemos la solución que ha sido aplicada para resolver este problema en la Sección siguiente.

Ontología base y ontologías extendidas

Ya hemos definido la ontología que almacena el conocimiento del dominio en el que opera nuestro sistema. Sin embargo, hemos de recordar que el programa que se ha diseñado y desarrollado no está orientado a un sólo tipo de narraciones, sino que puede crear contenido textual a partir de varias fuentes de datos, con, posiblemente, diferentes símbolos que tratar y diferentes significados. Si las fuentes de datos provienen de un sistema de diálogo entre robots autónomos, no vamos a disponer, muy probablemente, de un vocabulario que haga referencia a ideas filosóficas, por lo que carece de utilidad tener una sola ontología que aglutine el conocimiento de todos los sistemas. Por un lado, porque el trabajo sería inmenso para los sistemas que conocemos, y, por otro, porque habría que remodelar la ontología completa para cada nuevo dominio que quisiéramos aceptar. Hemos adoptado una aproximación que nos permite adaptar el sistema según sea necesario.

El uso que hemos dado a las ontologías consiste, por un lado, en la creación de una ONTOLOGÍA GENERAL que nos permite definir los conceptos básicos sobre los que siempre va a trabajar nuestro sistema. Estos conceptos están explicados en la Sección anterior, y básicamente conforman la raíz del árbol de clases de objetos, elementos y hechos que se pueden dar en las historias. Esta ontología es fija, y el sistema siempre la considera.

Pero, además, por otro lado, hemos hecho que para cada dominio para el que el sistema tenga obligación de generar historias, se ha de crear una extensión de esta ONTOLOGÍA GENERAL, de modo que establezca los tipos de clases más particulares de dicho dominio, o propiedades sobre esas clases más especializadas. Así, tendríamos, como entrada al sistema, el conjunto de datos como conocimiento añadido sobre el conocimiento que ya se encuentre en las ontologías base y extendida. Éstas, por lo tanto, representarían el "conocimiento compartido" que tanto el lector como el generador de historias tienen y que no ha de ser denotado explícitamente. Por ejemplo, si decimos:

Sócrates se sentó en la silla.

Consideramos que el lector ya sabe qué es una "silla", y no tenemos que explicarlo en la historia. Por tanto, en cada ontología que extiende la ontología base tenemos la posibilidad de almacenar un conjunto de instancias, no sólo de clases, que definen de antemano un conocimiento dado, que no ha de ser necesariamente escrito de manera explícita en los datos de entrada para cada historia.

Hasta aquí, sabemos que las ontologías complementan la información de la entrada de la historia. Sin embargo, se permite en el sistema la posibilidad

de *sobreescribir* la información que hay en la ontología con la información presente en la historia.

Cada símbolo de INSTANCIA de un elemento del mundo está únicamente definido por su identificador o índice. Por ejemplo, en la Figura 3.6 podemos ver una instancia de la clase ENTIDAD (ENTITY en inglés).

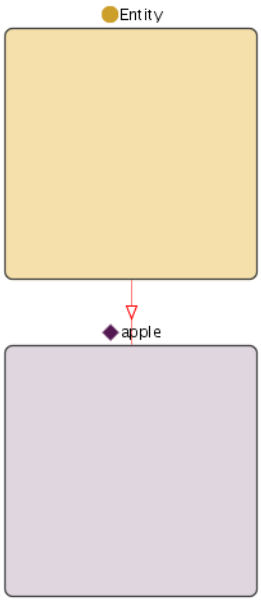

Figura 3.6: Ejemplo de una instancia con nombre.

Pero podemos, en la historia, sobreescribir esta información. Si quisiéramos, por ejemplo, que "apple" fuera un LUGAR y no una ENTIDAD, simplemente tendríamos que escribir, en un archivo de historia, la nueva información, y el sistema automáticamente reescribirá en su representación interna la información (Figura 3.7). Hay que tener en cuenta que la inconsistencia que se pudiera ocasionar de esta reescritura no es considerada por el sistema, y el usuario es el único responsable de que se cumpla la coherencia semántica en el dominio de la historia.

```
# Example
location apple
```

Figura 3.7: Un posible cambio de tipo de REFERENTE en la historia, sobreescribiendo el que existía en la ontología.

3.1.2. Especificación de los datos de entrada

Lo necesario para que el sistema generador comience a funcionar, pues, es disponer de una interfaz de comunicación. En este prototipo se han considerado tres alternativas: un formato creado para esta aplicación, el uso de XML y la interacción nativa con JAVA. Lo necesario es que los datos que se originan en los sistemas de múltiples personajes puedan ser transferidos a la aplicación de generación de texto de una forma estructurada, de modo que la aplicación pueda interpretarlos, y pasen así de ser simples datos a información. A continuación damos más detalle de estas alternativas propuestas.

Comunicación con ontologías

Evidentemente la manera más directa de crear las instancias de nuestros PERSONAJES, LUGARES y demás es la de insertarlos directamente en la ontología del dominio. Simplemente, usando un editor de ontologías, o trabajando directamente sobre el código en OWL es posible crear todas las instancias necesarias del dominio de la historia. La Figura 3.5 representa un ejemplo de estas ideas.

No obstante, ésta, desde el punto de vista del trabajo de investigación que presentamos, no es la mejor alternativa. Incluyendo la información en la ontología conseguimos un sistema muy rígido, en el que la aplicación que genera datos tiene que modificar la ontología (aunque sólo las instancias) para luego alimentar el generador de historias. A continuación presentamos otras alternativas que consideramos más cómodas.

Comunicación con formato propio

Existen varias maneras, como es evidente, de generar datos para alimentar al sistema de narración. Una de ellas es aprovechar aplicaciones o sistemas, como se explica en la Sección 4.2. No obstante, no siempre es posible disponer de tales aplicaciones, ya que en ocasiones es muy costoso desarrollar un sistema de varios personajes más o menos inteligentes, y puede no merecer la pena si la única finalidad de éstos es alimentar los datos de un prototipo de generación de historias en lenguaje natural.

Por esta razón, se ha creado un formato de archivo de datos para el sistema, en el que se pueden especificar, en un simple archivo de texto, los eventos que se suceden en un dominio dado, para que el generador de historias pueda importarlo y crear con él una representación del mundo, como veremos más adelante, en la Sección 3.2.1.

Este formato, tal y como hemos comentado antes, simplemente consta de una lista de hechos que conforman el conjunto de datos del *mundo* del generador, entendiendo como *mundo* todos los datos a partir de los cuales la historia puede ser creada.

A continuación, en la Figura 3.8, se puede ver un ejemplo de este archivo. Contiene un fragmento del *Menón*, el diálogo platónico, que usamos durante toda la explicación del sistema. Los eventos aquí añadidos han sido deducidos a partir de la lectura del diálogo. Vemos en este ejemplo que tenemos *tipos de referentes* que corresponden a la ontología: VERBOS, como "be", LUGARES, PERSONAJES, etc. Además, aquellos referentes más complejos, como las PROPIEDADES, tienen ciertos campos que han de ser establecidos, y que corresponden a relaciones entre varios referentes, como saber el "quién" o "por qué" de un hecho determinado.

```
verb be
verb be_like
attribute mortality
attribute male
character socrates
character meno
entity 1
entity 2

property pr4 {who socrates; verb be; attribute male;
              init 1; end 2}
property pr3 {who meno; verb be; attribute male;
              init 1; end 2}
property pr1 {who meno; verb be; attribute mortality;
              init 1; end 2}
property ev1 {who socrates; verb be_like; attribute
              meno;init 1; end 2}
property pr2 {who socrates; verb be; attribute
              mortality;init 1; end 2; because
              pr1, ev1;}
```

Figura 3.8: Ejemplo de un archivo de entrada para el sistema, con la representación de un fragmento de un diálogo de Platón.

Comunicación con XML

El formato propio que se ha creado y acaba de ser expuesto tiene la característica de ser muy sencillo de aprender, y los documentos, las historias generadas con él se escriben con mucha facilidad. Sin embargo, este formato ha sido creado sólo para crear historias de manera manual, no para comunicar sistemas de personajes con nuestro sistema narrador. Para esto se ha optado por añadir otra posibilidad: usar XML, con sus propiedades de estructuración, como sistema de comunicación.

La expresividad de ambos sistemas es exactamente la misma, pero usar XML tiene ciertas ventajas, como explicamos a continuación. El único incon-

veniente respecto del sistema de generación anterior es la mayor dificultad para generar documentos, ya que la sintaxis de XML es algo engorrosa. Sin embargo, las aplicaciones pueden realizar esta tarea de modo transparente, así que, disponiendo de ambas alternativas, eliminamos ambos problemas.

El la Sección 2.4 se ha presentado el lenguaje XML como un lenguaje de marcado que ha ido convirtiéndose con el uso en un estándar de facto para la comunicación entre plataformas, debido a sus características. Siguiendo estas ideas, y teniendo en cuenta que uno de los requisitos fundamentales de esta investigación es poder unir el generador de historias con varias fuentes de datos, se impone la necesidad de disponer de un sistema capaz de comunicar sistemas informáticos de diversos tipos.

Por tanto, se ha optado por ofrecer la posibilidad de usar XML para la comunicación. Así, un programa escrito en C/C++ sólo tendría que crear un archivo XML con el formato y la DTD adecuada. A pesar de que las estructuras de datos que se exportan podrían ser representadas con cualquier formato, se ha decido usar XML. La razón principal es la amplia disponibilidad de herramientas existentes para aprovechar la potencia de este estándar, las capacidades de estructuración sencilla, y la posibilidad de evaluación de la corrección y la buena formación de los documentos integrada en el estándar XML. La DTD se puede ver en la Figura 3.9.

La estructura que los archivos XML han de tener ha evolucionado a partir de los primeros prototipos de la aplicación. El desarrollo y el refinamiento de la gramática documental ha ido paralelo al progreso de los sistemas que generan datos que han sido usados. El objetivo principal ha sido deducir una plantilla DTD que cumpliese de forma genérica los requisitos de cada uno de los sistemas que generan datos de personajes. Así, a pesar de que la sintaxis gramatical resulta más grande de lo necesario para sistemas concretos, conseguimos un único formato de entrada que todos los programas que generan datos de ejecución de personajes deben seguir cuando generan el archivo de salida en XML.

En la Figura 3.10 mostramos un ejemplo que consiste en un fragmento de un XML de entrada.

Comunicación de modo nativo

La interacción basada en XML o el formato propio del generador que se ha creado es general y cómoda, pero no es rápida. Un programa nativo en Java debería ser capaz de llamar al generador de historias directamente en la Máquina Virtual. Para este objetivo, nuestra aplicación ha sido empaquetada en un archivo JAR que puede ser usado como libería de forma nativa.

De este modo, cualquier programa Java puede disponer de nuestro generador de contenidos, y podría ser usado para varias aplicaciones, y de muy distintos tipos, de una manera muy sencilla. Simplemente examinando

```
<!ELEMENT Attribute EMPTY >
<!ATTLIST Attribute Id NMTOKEN #REQUIRED >

<!ELEMENT Character ( #PCDATA | Event | Property )* >
<!ATTLIST Character Id NMTOKEN #REQUIRED >

<!ELEMENT Description ( #PCDATA ) >

<!ELEMENT Entity EMPTY >
<!ATTLIST Entity Id NMTOKEN #REQUIRED >

<!ELEMENT Event EMPTY >
<!ATTLIST Event Because NMTOKEN #IMPLIED >
<!ATTLIST Event Direct NMTOKEN #IMPLIED >
<!ATTLIST Event End NMTOKEN #REQUIRED >
<!ATTLIST Event Id NMTOKEN #REQUIRED >
<!ATTLIST Event Indirect NMTOKEN #IMPLIED >
<!ATTLIST Event Init NMTOKEN #REQUIRED >
<!ATTLIST Event Verb NMTOKEN #REQUIRED >
<!ATTLIST Event Where NMTOKEN #IMPLIED >
<!ATTLIST Event How NMTOKEN #IMPLIED >

<!ELEMENT Property EMPTY >
<!ATTLIST Property Because NMTOKEN #IMPLIED >
<!ATTLIST Property End NMTOKEN #REQUIRED >
<!ATTLIST Property Id NMTOKEN #REQUIRED >
<!ATTLIST Property Init NMTOKEN #REQUIRED >
<!ATTLIST Property Value NMTOKEN #REQUIRED >
<!ATTLIST Property Verb NMTOKEN #REQUIRED >

<!ELEMENT Story ( Description, Entity*,
          Attribute*, Verb*, Character* ) >
<!ATTLIST Story Id NMTOKEN #REQUIRED >

<!ELEMENT Verb EMPTY >
<!ATTLIST Verb Id NMTOKEN #REQUIRED >
```

Figura 3.9: DTD de los XML de datos.

el interfaz de llamadas de la aplicación (API) puede conseguirse un enlace sencillo y rápido.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Story Id="meno">
  <Description>
    Meno and Socrates
  </Description>

  <Entity Id="athens" />
  <Attribute Id="male"/>
  <Attribute Id="mortality"/>
  <Verb Id="be"/>
  <Verb Id="go"/>
  <Verb Id="say"/>

  <Character Id="meno">
    <Property Verb="be" Id="pr4" Init="1" End="2"
     Value="male" />
    <Property Verb="be" Id="pr3" Init="1" End="2"
     Value="mortality" />
    <Event Id="ev1" Init="3" End="3" Verb="go"
     Where="athens" Because="ev7"/>
    <Event Id="ev2" Init="1" End="2" Verb="say"
     Direct="s1" Indirect="socrates"/>
  </Character>

  ...
</Story>
```

Figura 3.10: Ejemplo de XML de entrada de datos a la aplicación.

3.2. Proceso de los datos: Generaración de historias

Hasta aquí se han expuesto las estructuras y las ideas principales que se usan en este sistema de generación de historias en lenguaje natural. En esta sección vamos a describir el algoritmo con el que es crean, a partir de los datos de entrada, la selección y orden de los hechos y relaciones entre ellos que, finalmente, conformarán la historia final.

El algoritmo que se ha creado consiste en una búsqueda en el espacio de estados que genera historias posibles a partir de los datos de entrada de la vida de varios personajes. El algoritmo va añadiendo posibles sucesos a la historia, relacionados con los anteriores, y evalúa la similitud entre la historia encontrada y el objetivo del usuario. Tras explorar todo el árbol de soluciones válido, elige la historia cuyas características son más similares a lo que el usuario del sistema requería.

3.2.1. Objetivos: Representación formal de los datos para el sistema de narración

Los ordenadores necesitan una representación formal de los datos con los que se les alimenta para poder procesarlos y generar una salida que nos sea útil. Hasta aquí hemos hablado de la entrada como conceptos de una ontología, que pueden ser trivialmente traducidos a una representación informática, y ahí a algún lenguaje concreto de ordenador en el que especificar el comportamiento. Por tanto, la entrada del sistema, desde un punto de vista computacional, queda explicada en la Sección 3.1.2.

Las herramientas que, como la que se presenta en este trabajo, están orientadas a la comunicación entre el contenido de información almacenado en una máquina y un humano que va a recibirlo, no pueden soslayar el problema de la interacción entre el programa y el usuario.

Como hemos comprobado hasta aquí, la generación de texto en lenguaje natural a partir de trazas de ejecución es un proceso complejo que requiere mucha información adicional no presente en los datos de entrada, y que debe, por tanto, ser creada por la máquina (Sección 2.3), o incluida en el algoritmo de proceso por el humano.

Por tanto, la labor del hombre en la generación de historias no es trivial. Hemos de saber qué tenemos que generar como salida, y de qué representaciones internas disponemos para hacer funcionar el algoritmo en una máquina.

Si se consiguiera realizar una representación general de lo que quiere el usuario sobre la generación de textos, habríamos conseguido modelar los deseos de un humano, y eso, hoy por hoy, no es posible. Por esta razón, simplemente adoptamos una postura directa y especificamos mediantes símbolos dependientes del dominio qué queremos representar, y el algoritmo, mediante reglas que "entiendan" los deseos del usuario mediante la interpretación computacional (el proceso del programa), guiará la generación de la historia para que se ajuste lo más posible a lo que la persona que está utilizando el programa reciba la salida que desea recibir.

A cada uno de estos símbolos les hemos llamado Objetivos. Un Objetivo es, por tanto, una estructura que contiene:

- Información sobre **qué quiere** el supuesto lector saber sobre la historia.
- **Algoritmos** para conseguir, a partir de la intención del lector, conseguir un conjunto de mensajes que transmitirle.

En nuestro sistema, hemos creado un sistema de Objetivos basado en la herencia, usando su funcionalidad semántica de especificación de comportamientos. Así, tenemos, por ejemplo, el Objetivo del deseo del lector de conocer una característica cualquiera sobre un personaje y, por herencia de

este OBJETIVO, aquél cuyo significado establece que el lector desea saber el valor de la característica "nombre" de cierto personaje. Este comportamiento, por tanto, es una versión más específica del anterior.

Cada objetivo, por tanto, tiene una serie de parámetros que lo definen y unos algoritmos de reducción a mensajes legibles que conforman el discurso, como explicaremos a continuación, que hacen que tengan un comportamiento en la traducción similar a SCHEMAS.

3.2.2. Creación de los objetivos

El conjunto de OBJETIVOS que creamos no ha salido, por supuesto, de la nada. Es una parte fundamental del modelo de nuestro sistema y radicarlo en una base fuerte y coherente le confiere una solidez que es indispensable para que el sistema no presente lagunas como algoritmo y pueda, también, ser ampliado y modificado. Por tanto, los objetivos que hemos creado y presentamos en este trabajo ha sido extraidos del diálogo sobre el cual, como ejemplo, mostramos el comportamiento del sistema.

Para realizar este cometido se ha partido del diálogo "Menón" de Platón, utilizándolo como *corpus* o documento original de datos, para realizar técnicas de ingeniería inversa a partir del escrito. La idea ha sido identificar qué objetivos se han intentado satisfacer en el diálogo e ir anotándolos o extrayéndolos de forma jerárquica sin ningún método concreto.

La razón de no usar ningún algoritmo manual concreto para la extracción de los objetivos se debe a que no existen modelos de la perspectiva de los objetivos en los textos [BIR04, RD00, Tab06]. Esto quiere decir que no hay ninguna tabla de objetivos que podamos simplemente usar para identificar qué se pretende obtener en la percepción del lector cuando el autor escribe algo de una manera exacta. Muy probablemente, tener este modelo no sea posible, dadas las características de la producción narrativa humana. Sin embargo, del mismo modo que al leer un diálogo de Platón es posible entender la idea que quiere transmitir, también es posible identificar por objetivos lo que estamos entendiendo, y de este modo extraer el contenido que necesitamos.

Huelga decir que la extracción que se haga es de manera muy grande, subjetiva, y que cada autor de esta tarea obtendrá objetivos que pueden ser radicalmente opuestos. Para comprobar este hecho sólo hay que examinar la multitud de diversas opiniones sobre lo que el autor de un texto concreto existen, y las disputas sobre ellas que en infinidad de ocasiones se dan sin solución.

Ejemplo de la obtención de objetivos

Vamos a exponer un ejemplo clarificador de una extracción de los objetivos a partir del *corpus* del diálogo Menón de Platón que se ha realizado.

En este ejemplo mostramos un fragmento del diálogo, y explicamos cuál es el proceso de extracción. En el siguiente texto está escrito. Se refiere al momento en el que Sócrates le pide a Menón que le explique lo que pensaba Gorgias sobre la virtud.

> *Sócrates: No tengo buena memoria, Menón, y por lo tanto no puedo decirte ahora lo que pensé de él entonces. Y me atrevo a decir que él sabía, y que tú sabes lo que dijo: por favor, por lo tanto, recuérdame lo que dijo o, si prefieres, cuéntame tu punto de vista, porque sospecho que él y tú pensáis cosas parecidas.*

En este párrafo podemos ver que existe un objetivo principal que es el de pedir al interlocutor que le cuente algo. No obstante, podemos leer claramente que este objetivo no ha sido comunicado con una simple orden imperativa o enunciativa de deseo, sino con un párrafo entero cargado de elementos con semántica particular dependientes del objetivo concreto.

Ahora, analizando más el texto, vemos que podemos dividir el objetivo en varios. Como hemos comentado, no disponemos de un modelo, pero de una manera intuitiva, y siguiendo algunas pautas que nos dan las teorías sobre los objetivos del texto [MT88], podemos encontrar subdivisiones. Así, dividimos el objetivo principal de querer saber lo que dijo Gorgias en:

- Explicar a Menón que Sócrates no tiene buena memoria.
- Suponer que Menón tiene la respuesta.
- Pedirle esta respuesta.

Así, ya tendríamos más objetivos que compondrían el principal. Pero, aun habiendo dividido ya el problema, no lo tenemos solucionado, puesto que estos objetivos no son lo suficientemente concretos para ser considerados como *atómicos* (no descomponibles), y debemos reducirlos aún más. La siguiente lista muestra cómo hacemos la división:

- Explicar a Menón que Sócrates no tiene buena memoria.
  - Decir que no tiene buena memoria.
  - Explicar que esto es la causa de no poder responder.
- Suponer que Menón tiene la respuesta.
  - Decir que o Gorgias tiene una idea,
  - o Menón tiene la idea.
- Pedirle esta respuesta.

- Pedírsela explícitamente.
- Explicar que se la pide porque supone que la sabe.

Vemos que tenemos una división más importante en este punto. Ahora tenemos seis elementos de objetivo, y podemos considerar que son lo suficientemente concretos para buscar en la base de conocimiento que tenemos, e intentar sustituirlos. Si podemos hacerlo, habremos logrado el objetivo, y habremos conseguido generar un texto que corresponde a este párrafo del Menón. En la Figura 3.11 se representa de manera gráfica el proceso de extracción que hemos realizado. Vemos en ella que se ha llevado a cabo la abstracción de los objetivos, quitando de ellos la componente particular de lo que se dice. Con esto ya tenemos un nuevo esquema de traducción de objetivos, y cada vez que nos encontremos un objetivo como el primero, sabremos cómo podemos sustituirlo por contenido de la base de conocimiento.

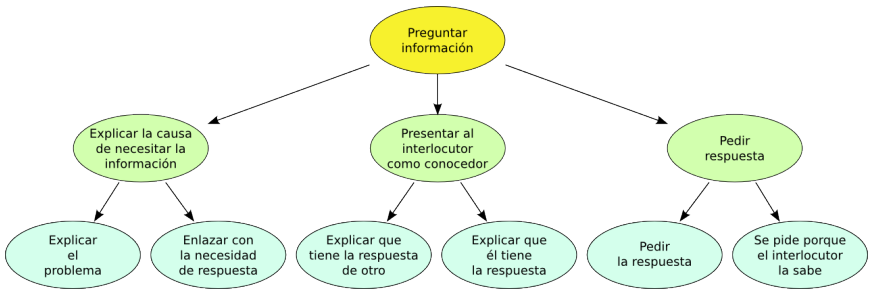

Figura 3.11: Representación de la extracción de los objetivos a partir de los diálogos de Platón.

En la figura, además, vemos las relaciones entre los objetivos, que son fundamentales. Las relaciones hacen que el lector tenga una visión más coherente del texto al entender, gracias a ellas, cómo se han desarrollado los hechos que se cuentan en función de los otros. Las relaciones que hemos usado para los objetivos son un subconjunto de la RST (Sección 2.1.7). Cada relación retórica de este conjunto está orientada a explicar los objetivos que ha establecido el escritor con respecto al texto escrito, por lo que es en este punto donde se muestra su potencial en relación con nuestra investigación.

Es fundamental notar que, como es más que habitual en los programas de planificación del discurso, y como ha sido presentado en la Sección 2.1.4, los Objetivos que creemos, al modelar el sistema, son precisamente parte del sistema, y no pueden estar, por la naturaleza con la que los construimos, desligados del dominio del mismo.

Por tanto, nuestros objetivos *son dependientes del sistema de generación.* Lo único a lo que podemos aspirar, siguiendo como hemos seguido estas ideas de generación, es que el sistema al que está acoplado nuestro algoritmo sea uno general que englobe los particulares sobre los que queremos generar

historias, de modo que alcancemos una altura más global en la generación de historias en lenguaje natural. Estas ideas las discutimos en la Sección 4.5.2

3.2.3. Traducción de objetivos: Planes de discurso

Una vez que disponemos de un árbol de objetivos del discurso generamos como acabamos de ver en la Sección 3.2.2, debemos traducirlos a un texto que represente una historia en lenguaje natural. Para ello, una vez que hemos llegado a un punto en el que cada objetivo es reducible según nuestro modelo, podemos buscar en la base de conocimiento que hemos adquirido para el sistema como contamos en la Sección 3.1, y crear un árbol documental, que es el principal objetivo de este proyecto.

Desde luego es necesario disponer de una estructura fija para crear la salida del sistema. En el nuestro se ha elegido adoptar una de las aproximaciones más clásicas en la GLN: los Planes de discurso. Estas estructuras consisten en datos que almacenan cierta información para el lector (un hecho o un evento) pero relacionadas con las demás según un esquema intencional, siguiendo las teorías de la RST [MT88].

Definimos Planes de discurso como un conjunto finito de estructuras de datos que, una vez desplegadas como discurso completo, adquieren forma de un árbol documental, en el que cada nodo posee, además de información por sí mismo, una relación determinada en la que queda clara su intención con respecto al resto del texto. En la Figura 3.12 podemos ver la jerarquía de los Planes de discurso con la que trabajamos en esta investigación.

En la figura representamos los planes que hemos diseñado como una jerarquía de clases, pues se ha considerado que esta representación era la que de manera más clara iba a mostrar la forma que tiene nuestra aportación. En la lista que sigue damos más detalle.

- **Unidad semántica** (SemanticUnit). Se ha optado, de la misma manera que hacemos en la ontología con los Referentes, por crear una unidad cuya intención sea la de agrupar a todos aquellos elementos que puedan contener un significado por sí mismos, y que puedan aparecer en el discurso final. Son las Unidades semánticas. Se dividen en dos clases de elementos:

  - **Plan de discurso** (DocumentPlan). Los planes del discurso son aquellas Unidades semánticas que son divisibles en más Unidades semánticas, sean de la naturaleza que sean. Es posible crear diferentes tipos de esta clase, pero se ha trabajado en esta aportación con Planes de discurso como tales, y con una subdivisión:

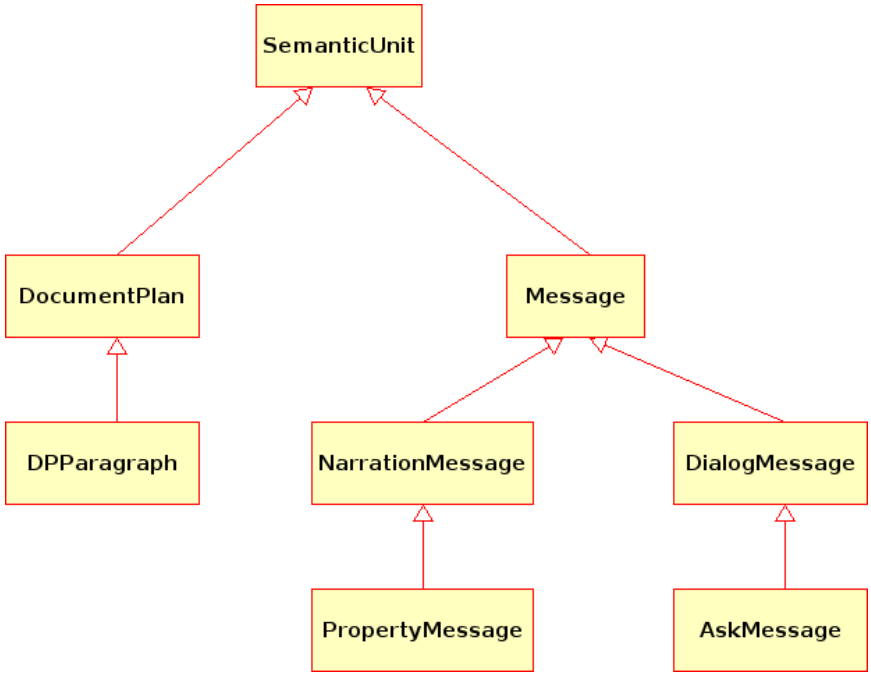

Figura 3.12: Figura que representa la estructura que usamos de los PLANES DE DISCURSO.

  - **Párrafo** (DPPARAGRAPH). Cada PÁRRAFO es una unidad de significado que, como su propio nombre indica, corresponde a un conjunto de UNIDADES SEMÁNTICAS agrupadas y ligeramente aisladas, por su significado, relación o localidad, de las demás de un discurso. En resumen, va a corresponder a un párrafo en el texto final.

- **Mensaje** (MESSAGE). Los MENSAJES son UNIDADES SEMÁNTICAS que ya sí corresponden a un elemento relacionado con la base de conocimiento (en general, identificado con uno de ellos), y que puede ser expresado de forma independiente del discurso, a pesar de que si él seguramente carecerá de coherencia para un lector. Los MENSAJES pueden ser de tipo *narrado* o de tipo *diálogo*. Esta división es fundamental en nuestro sistema, ya que no se tratarán igual, como veremos, los dos tipos de estructuras.
  - **Mensaje Narrado** (NARRATIONMESSAGE). Para cada tipo de información que se quiere expresar se ha creado un tipo de mensaje concreto, que no reproducimos aquí por brevedad y concisión. Por ejemplo, para decir "Menón quiere tener una respuesta", crearíamos un mensaje de tipo *Deseo*, en el que guardaríamos la información concreta sobre el deseo. La cantidad y los tipos de mensajes que se crean son muy

dependientes del dominio. Un ejemplo de mensaje narrado es un mensaje de PropertyMessage, que expresa un atributo de un personaje.

- **Mensaje de Diálogo** (DialogMessage). Estos mensajes son similares a los anteriores, con la diferencia de que su contenido es algo que se ha dicho. En ellos es importante saber quién ha sido el autor, y el orden en la conversación es determinante. AskMessage, que expresa una sentencia en forma interrogativa, podría ser un ejemplo de un mensaje de diálogo.

Con estas ideas sobre la salida del sistema en la Sección 3.3 vamos a explicar cuál es la salida global del sistema. Sin embargo, aún tenemos simplemente Objetivos como deseo de traducción, y Planes de discurso como producto de la misma, queda por definir algoritmo que realizará la traducción.

Cuando tenemos los objetivos definidos, es posible realizar una operación de *traducción* sobre ellos, de modo que convirtamos el deseo que, según nuestro modelo, tiene el lector, a una estructura legible de discurso. Para este propósito se ha dotado a los objetivos, como hemos avanzado, de algoritmos que realizan estas operaciones. La Figura 3.13 esquematiza estas ideas. En el dibujo se ve que un objetivo posiblemente compuesto de subobjetivos es traducido a un discurso posiblemente constituido por subdiscursos. Esta operación es la base de nuestro generador de historias.

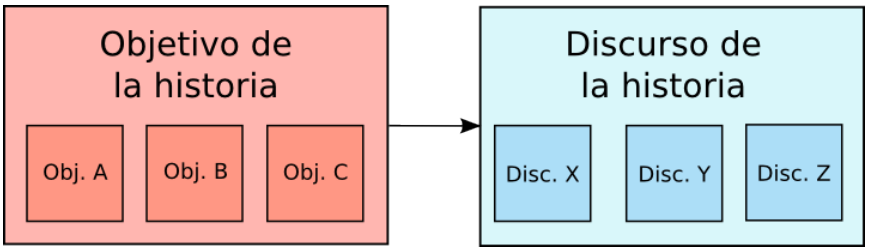

Figura 3.13: Esquema que representa la traducción de un Objetivo a un Plan de discurso.

El algoritmo con el que se realiza la traducción consiste en una *búsqueda recursiva en profundidad* sobre *esquemas de discurso* guiada por la *percepción del lector del discurso narrativo*. Vamos a definir y explicar estas ideas en los subapartados siguientes.

Esquemas de discurso

En la Sección 3.2.2 hemos visto cómo conseguir objetivos mínimos a partir de objetivos más generales y posiblemente complejos. En esta sección

vamos a ver cómo traducir estos objetivos ya reducidos a planes de discurso apropiados.

Hemos seguido las ideas clásicas, como hacen muchos autores, de los SCHEMAS [McK85]. El concepto es partir de una representación que simbolice un estado particular de la generación, e intentar traducirlo por mensajes de forma que, dado un objetivo, se devuelva un discurso. Para esto usamos todo un sistema de producciones que se van a aplicar a objetivos concretos según el algoritmo que explicamos en la Sección 3.2.3. Dichas producciones tienen una estructura de gramática independiente del contexto, en la que los terminales son conjuntos de mensajes y los no terminales son objetivos.

Es importante notar que las producciones, es decir, los pasos desde los objetivos a los conjuntos de mensajes, en ocasiones, para una regla concreta, tiene la misma precondición, y diferentes postcondiciones, es decir, un objetivo puede traducirse por varios conjuntos de mensajes distintos. Qué regla de producción elegir va a ser determinado por la base de conocimiento y la percepción del discurso narrativo.

Siguiendo las producciones, pues, creamos, según vamos aplicándolas, una traducción, de una manera parecida a la siguen generalmente los compiladores [ASU88]. Con cada producción tenemos un discurso parcial que, al ascenderlo por el árbol gramatical y unirlo según las reglas que se han ido escribiendo para definir la gramática, nos dan el discurso final.

A continuación mostramos un ejemplo de la traducción que puede servir para aclarar un poco más las ideas que hemos comentado de forma general hasta aquí. En este ejemplo vemos como reducimos un objetivo usando la gramática hasta tener un discurso parcial concreto.

Tenemos, por ejemplo, el objetivo de REASONGOAL, que tiene como propósito, y así está programado en él, razonar las causas de un hecho dado a partir del conocimiento que tiene el sistema. Este objetivo crea un discurso con un solo párrafo que tiene como núcleo el hecho que queremos razonar, y como precondiciones todas aquellas que encuentre en la base de conocimiento. En la Figura 3.14 podemos ver un ejemplo de la ejecución de la aplicación que se ha programado.

Este ejemplo podría traducirse, para que quede más clara la expresión, como:

> *Si Menón es mortal y Sócrates es como Menón, entonces Sócrates es mortal.*

Al reducir el objetivo se han tenido en cuenta varios parámetros que comentamos más adelante. El que tiene relación con esta parte de la memoria del trabajo de investigación es el de la extracción de la base de conocimiento de los hechos indicados. Para realizar este cometido, cada objetivo tiene un algoritmo programado según unas reglas determinadas que son precisamente la semántica del objetivo, y se encargan de acceder a la base de

```
Content Planner - (c) Carlos León 2007.

Creating world "logic.txt"...
Added "verb" with id "be". Ok.
Added "verb" with id "be_like". Ok.
Added "attribute" with id "mortality". Ok.
Added "attribute" with id "male". Ok.
Added "character" with id "socrates". Ok.
Added "character" with id "meno". Ok.
Added "entity" with id "1". Ok.
Added "entity" with id "2". Ok.
Added "property" with id "pr4". Ok.
Added "property" with id "pr3". Ok.
Added "property" with id "pr1". Ok.
Added "property" with id "ev1". Ok.
Added "property" with id "pr2". Ok.
Creating table links...
World created succesfully.
Created ReasonGoal for fact "pr2".
Discourse plan with quality 1.00:
[PARAGRAPH:
  [NS:
    [CONDITION:
      [CONJUNCTION:
        [meno] [be] [mortality].
        [socrates] [be_like] [meno].
      ]
    ]
    socrates be mortality.
  ]
```

Figura 3.14: Ejemplo de una ejecución del prototipo traduciendo un solo objetivo.

conocimiento. En función de qué objetivo sea, puede que sea capaz de encontrar hechos particulares para él, o no. Si falla y no puede "narrar" este objetivo, habría que encontrar otra opción. Esta búsqueda, como técnica de Inteligencia Artificial, se explica en la Sección 3.2.3.

Por ejemplo, éste que acabamos de exponer en la Figura 3.14 tiene como algoritmo de traducción el que se expone en la Figura 3.15, como lista de operaciones.

De este modo se pueden definir todos los objetivos, con su algoritmo de reducción.

1. Crear un párrafo
2. Crear, dentro del párrafo, una relación retórica:
   a) El núcleo será lo que se quiere justificar.
   b) Los satélites serán todos aquellos elementos de la base de conocimiento con los que el núcleo tenga una relación de consecuencia.
3. Si no existen los satélites, el objetivo falla.
4. Si existen, el objetivo es válido y se crean las relaciones de CONJUNCIÓN entre los satélites, y de CONDICIÓN entre los satélites y el núcleo.

Figura 3.15: Algoritmo del objetivo REASONGOAL

Búsqueda recursiva en profundidad

Hasta aquí se ha explicado cómo traducir un nombreObjetivo a una representación estructurada del discurso. Sin embargo, sabemos que existe una cantidad potencialmente infinita de posibilidades de narrar una historia, aun manteniendo una semántica, e incluso un orden de los hechos dado. Sin pretender, por supuesto, alcanzar un examen de todas las posibilidades (cosa imposible), nuestro sistema realiza un búsqueda entre muchas posibilidades de contar la historia, eligiendo la mejor posible.

Para crear esta búsqueda se crea un espacio de posibles opciones sobre que se explora para encontrar la mejor según unos términos que más adelante explicamos. Esta búsqueda se plantea siguiendo las técnicas clásicas de la Inteligencia Artificial de búsqueda en el espacio de estados, desde un estado inicial hasta uno suficientemente bueno que será elegido como la solución del problema, en este caso, el discurso más apropiado.

La primera aproximación tomada para buscar en el espacio de estados ha sido realizar una búsqueda en anchura común para explorar el árbol de soluciones. Esta alternativa, sin embargo, es muy ineficiente: el número de nodos de búsqueda que corresponden a historias parciales es exponencial, con lo que el coste en términos de memoria y tiempo es muy elevado. Sabemos, por otro lado, que en nuestro caso la búsqueda termina si programamos bien las reducciones de los OBJETIVOS, con lo que el algoritmo es completo y óptimo en términos de la heurística con la que valoremos los discursos, de cuya optimalidad no tiene sentido hablar, ya que la narración no tiene una valoración absoluta posible. Por tanto, tiene sentido usar esta técnica.

El coste es tan elevado que no han podido llevarse a cabo pruebas reales, ya que la memoria de la máquina utilizada[1] no era suficiente para un colectivo de 10 personajes con 10 hechos cada uno. En la implementación que se ha

[1] 768 Megabytes de memoria RAM

realizado, en la máquina anterior, la MÁQUINA VIRTUAL DE JAVA lanzaba una excepción de tipo `OutOfMemoryError` (error por desbordamiento de la memoria), por lo que nunca se ha podido llegar a finalizar una ejecución.

Por lo tanto, la solución para resolver este asunto, dados los problemas de memoria, fue traducir el algoritmo de búsqueda en el espacio de estados, transformándolo de uno en anchura, basado en una *cola* por uno en profundidad, basado en una *pila*. Los algoritmos basados en una pila tienen el mismo coste asintótico en tiempo, pero no en memoria. Gracias a este cambio, la MÁQUINA VIRTUAL DE JAVA no tiene problemas de desbordamiento de memoria, y el algoritmo puede terminar. Evidentemente, pese a haber resuelto los problemas de memoria, el coste en tiempo sigue siendo muy grande.

Si durante la búsqueda exploramos todos las posibles historias que pueden ser generadas, nuestro algoritmo es completo. Sin embargo, no significa que sea óptimo de manera general, recordemos, ya que la evaluación de la similitud entre la historia generada y el objetivo que ha de cumplir la historia, como explicamos en la Sección 3.2.4, es de carácter heurístico, y no asegura ninguna optimalidad. Más adelante se discuten estas ideas.

Siguiendo esta manera de generar discursos, el algoritmo crea todos los posibles siguiendo una búsqueda exhaustiva en profundidad. El algoritmo, pues, elige un primer objetivo que se establece a mano como entrada del sistema (lo que el lector debería querer saber), y construye varios discursos. La Figura 3.16 representa estas ideas.

Percepción del discurso narrativo por parte del lector

Hoy en día modelar el cerebro humano está fuera de nuestro alcance. Existe una gran cantidad de disciplinas de todos los ámbitos volcadas en el conocimiento de los mecanismos mentales de las personas. Desde luego, esta investigación no pretende crear un modelo de esa magnitud, pero si permitimos varias relajaciones, y nos centramos en los aspectos puramente literarios, y sólo, desde luego, funcionales, podemos conseguir un sistema que, a pesar de no emular al hombre, sea capaz de guiar la creación de historias.

A medida que el árbol de búsqueda de estados que representan historias apropiadas va siendo generado, en cada paso, como historia parcial, calculamos el estado del modelo del lector como perceptor de la historia. En esta sección explicamos cómo, a lo largo de la generación de la historia, la percepción del lector, según la hemos creado, va evolucionando.

La visión que el lector tiene de lo que se está contando, su imagen mental, no sólo es modificada por los atributos físicos de los personajes. En una historia que pretenda tener algo de contenido causal, es imperativo disponer de un mecanismo que describa el estado mental de los personajes, de modo que el lector pueda conocer las motivaciones, creencias y deseos de los

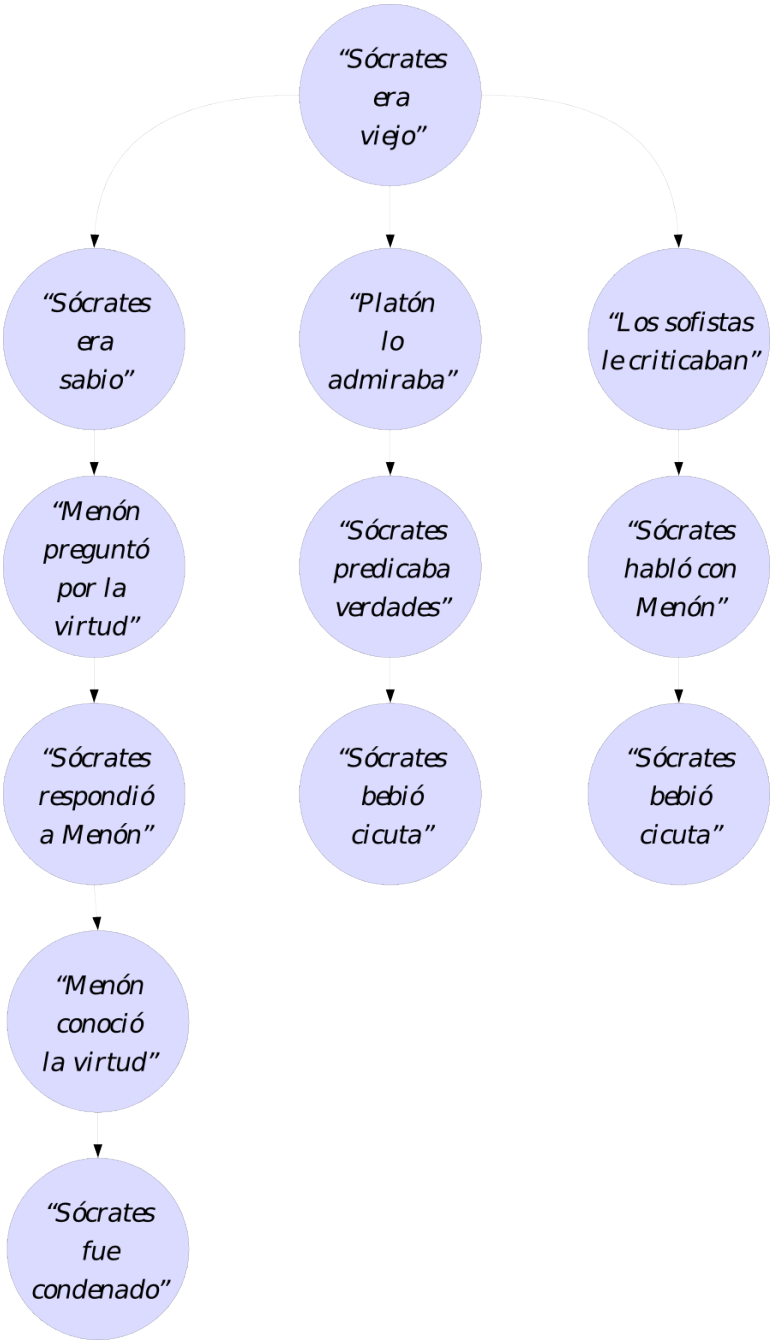

Figura 3.16: Árbol que representa la búsqueda con diferentes posibles historias encontradas.

mismos.

En este trabajo se ha implementado las ideas BDI (*Beliefs, Desires and*

Itentions) de agentes [Bra87] para modelar los personajes. Cada personaje en la historia, por tanto, de la misma manera que los agentes BDI, tiene un conjunto de *creencias*, *deseos* e *intenciones* que definen su estado mental.

Durante la historia, el sistema no crea la información BDI de los personajes, simplemente la extrae de los hechos que van añadiéndose al relato final. Es decir, la semántica de las reglas que gobiernan el modelo de los personajes ha de estar especificada en las aplicaciones que generen los datos de entrada del sistema.

La percepción del lector de los personajes sigue la historia, y tiene que ir cambiado según ésta avanza. Para dar un ejemplo explicativo de esta idea, si tenemos el siguiente texto (codificado en la entrada, desde luego, como datos de ordenador, no como texto legible):

Los sofistas criticaban a Sócrates.

Podemos decir que el personaje "Sócrates" tiene un conocimiento del mundo determinado acerca de lo que los sofistas piensan de él. De manera estructurada, la representación del personaje con este dato añadido sería la que se refleja en la Tabla 3.1.

Personaje	
Creencias	[`desprecio de los sofistas`]
Deseos	[$\cdots$]
Intenciones	[$\cdots$]
$\cdots$	$\cdots$

Tabla 3.1: Tabla que representa el conocimiento del estado mental de un personaje con *creencias*.

Si, además, el lector lee en la historia que nuestro personaje, "Sócrates", desea ser respetado por los sofistas, tendríamos que actualizar el estado mental del personaje "Sócrates" con el deseo que tiene de ser respetado. Si la frase fuera:

Sócrates deseaba ser respetado por los sofistas.

La representación sería la mostrada en la Tabla 3.2.

Finalmente, dados unos *deseos* y unas *creencias*, los personajes generan *intenciones*. Estas *intenciones* son aquellas decisiones del personaje sobre la acción a realizar. Con frases como:

Sócrates decidió beber cicuta.

Personaje	
Creencias	[`desprecio de los sofistas`]
Deseos	[`respeto de los sofistas`]
Intenciones	$[\cdots]$
...	...

Tabla 3.2: Tabla que representa el conocimiento del estado mental de un personaje con *creencias* y *deseos*.

Personaje	
Creencias	[`desprecio de los sofistas`]
Deseos	[`respeto de los sofistas`]
Intenciones	[`beber cicuta`]
...	...

Tabla 3.3: Tabla que representa el conocimiento del estado mental de un personaje con *creencias*, *deseos* e *intenciones*.

El estado de la percepción del lector acerca del personaje "Sócrates" se actualizaría, y daría lugar a una nueva estructura de datos como la representada en la Tabla 3.3. Esta tabla representa, evidentemente, un estado en el cual también se han incluido las frases anteriores.

Por lo tanto, vemos cómo, con estas ideas, la percepción del lector sobre los personajes de la historia puede ser modelada. Veremos, en el algoritmo que genera, ordenando y filtrando, los hechos de la historia final, que esta información es indispensable para conseguir una historia que transmita la información que se quiere transmitir a los lectores de una manera coherente.

Los personajes dentro de una historia no son simples entidades formales con *deseos*, *creencias* e *intenciones*, sino que representan seres con propiedades no sólo mentales más o menos complejas, dependiendo del tipo de historia o narración, y como tales su descripción no puede ser simplemente una referencia, es decir, "Sócrates" no es nada con significado hasta que no está descrito. Por ejemplo, en una historia del estilo de la siguiente, no sabemos quién es el Sócrates ni qué importancia tiene que no sea un sofista:

> *Sócrates no era un sofista.*

Sin embargo, si añadimos algunos atributos previos a la descripción de la acción, podemos entender mejor cuáles han sido las causas:

> *Había un filósofo que se llamaba Sócrates. Existían filósofos sofistas. Los sofistas enseñaban la areté. La areté era la virtud. Sócrates buscaba la virtud. Sócrates no era un sofista.*

Por esto, el conocimiento que tiene el lector de los atributos de los personajes es una muy buena parte del contenido que ha de serle transmitido, y por esta razón ha sido incluido en el modelo de la percepción del lector.

Para hacer esta inclusión, la estructura de datos lógica que representa el conocimiento del lector en un momento dado de la historia incluye una lista de los personajes de cuya existencia el lector tiene consciencia. De estos personajes "conocidos", guarda los atributos que se saben. Es decir, si en la historia ha aparecido algo del estilo de:

Había un filósofo que se llamaba Sócrates.

representado de manera estructurada, como se cuenta en la Sección 3.2.1, al añadir esa sentencia a la historia final, la percepción del lector se actualizará con la información la Tabla 3.4.

Personaje	
Nombre	`Sócrates`
Profesión	`filósofo`
Edad	?
Género	?
...	...

Tabla 3.4: Tabla que representa el conocimiento de los atributos de un personaje.

Y, por tanto, el lector, según el modelo, tendrá esta información, que servirá para conducir el algoritmo de generación de historias. Básicamente, podemos ver estas ideas resumidas y representadas gráficamente en la Figura 3.17. El texto correspondiente a este discurso sería:

Sócrates era viejo y sabio, por eso, los sofistas lo criticaban.

Podemos resumir hasta aquí que no disponemos de una descripción funcional del cerebro humano, pero pesar de eso, no es imposible nuestra tarea, ya que, por un lado, sólo *estamos modelando una pequeña parte del comportamiento humano*, y por otro *no es necesario que el modelo se adapte totalmente al comportamiento humano*. Lo más importante de la descripción de lo que piensa el lector que hemos hecho es que es una información adicional para la creación del discurso que podemos usar e ir actualizando para generar el discurso.

Para poner un ejemplo que aclare, la representación secuencial de la Figura 3.18 muestra cómo usar la percepción del lector en un caso concreto.

Como vemos, la idea de la percepción de la historia por parte del lector es más potente que la simple lista de referentes, ya que nos permite muchas más flexibilidad del discurso aplicando de manera no demasiado estricta el modelo BDI.

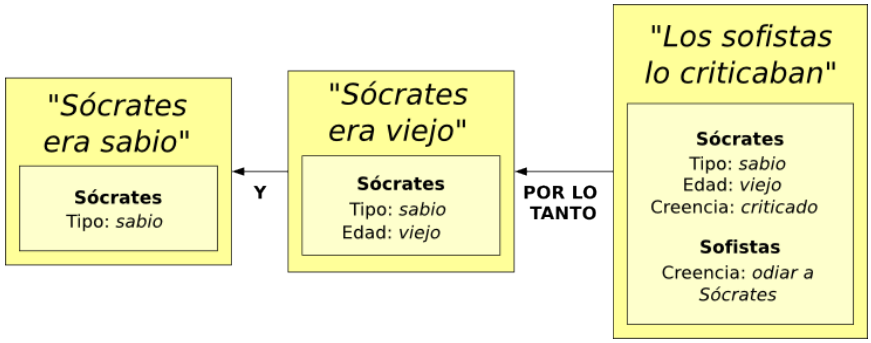

Figura 3.17: Discurso narrativo. Las cajas internas representan la percepción del lector.

1. Tenemos como OBJETIVO decir datos sobre los conocidos de un personaje.
2. Buscamos todas las relaciones en la base de conocimiento que pertenezca al conjunto de *relaciones entre personajes*, como "amigo", "enemigo", "padre", etcétera.
3. Comprobamos en la percepción del lector.
   a) Si alguno de los personajes referenciados ya se conocen, se desechan, y no se añaden.
   b) Si existe algún personaje que tenga intención de saber algo sobre alguno, se crea un OBJETIVO de relacionar éste personaje que aquél.
4. Creamos un discurso con los posibles OBJETIVOS creados y los MENSAJES SOBRE LOS OTROS PERSONAJES
5. Actualizamos la percepción de la historia del lector con todos los mensajes que hayan sido añadidos.

Figura 3.18: Ejemplo de la actualización de la percepción del lector para un caso concreto.

3.2.4. Evaluación de la traducción: creación de la heurística para las historias

Se ha comentado en la Sección 3.2.3 que se prueban diferentes posibles historias para luego conseguir una historia final que, según cierta heurística, se la más adecuada de todas. En este apartado vamos a explicar cómo se extrae dicha heurística y cómo se usa.

Muchos sistemas, como [PG06] emplean un sistema de valoración de los

textos externa a la generación y posterior a ella que consiste en evaluar de una manera o de otra los textos generados, dándoles una "nota" a cada una de sus partes (Sección 2.3.1). Después, se analiza la evaluación de los textos y se cierra el lazo aplicando las conclusiones en los retoques de algunas partes determinadas de la generación de la historia.

Nuestro sistema sigue una aproximación similar, sin embargo la técnica que hemos empleado consiste en incluir directamente las valoraciones en el texto, no retocando el algoritmo, sino ajustando parámetros de una heurística directamente desde la evaluación de los usuarios.

Uso de la heurística

Cada discurso parcial que generamos tiene implícitamente una *calidad*. Esta *calidad* es muy difícil de definir, y es, además, tremendamente subjetiva (¿quién era mejor poeta: Francisco de Quevedo, o Luis de Góngora?). Sin embargo, para crear un discurso narrativo mediante una búsqueda necesitamos un valor comparable (un número) que valore cada parte generada, de forma que podamos guiar y terminar la búsqueda.

Por tanto, necesitamos una función de valoración de un discurso, una función *heurística* que definamos de tal forma que sea posible saber el valor de la misma para cada discurso. Para esto se ha diseñado una función de varios términos que se calcula a partir de cada discurso. En la siguiente lista mostramos los valores que se han incluido en el prototipo:

- **Coherencia** (CH), o cómo ve el lector de coherente un texto.
- **Orden** (O), si sigue un orden lógico que un humano podría haber establecido.
- **Conocimiento** (CO), o cuánto sentido tiene referenciar elementos en esa parte según lo que se sabe de la historia.
- **Distribución** (D) de parte hablada y narrada de forma coherente.

Así, a cada discurso parcial que se genera se le puede aplicar esta función, y cada discurso parcial tiene unos valores particulares de estos parámetros, siendo forma, en consecuencia, la *heurística* de esta forma:

$$h(discurso) = \frac{CH + OR + CO + D}{4} \tag{3.1}$$

Así, en la búsqueda, sabremos calcular la "calidad" de las partes de la historia generadas, con lo que será posible realizar un busqueda correcta.

Extracción de la heurística

Queda explicar cuál es el proceso de averiguación de los valores de esta heurística. Para conseguir este propósito, que no es en absoluto sencillo, nos hemos basado en aproximaciones previas sencillas que están basadas en la encuesta de varios usuarios sobre textos concretos generados con la intención de que valoren el texto que se les da, y, sobre estas encuestas, realizar técnicas estadísticas básicas para extraer los valores que han de tener los parámetros. Para realizar la explicación vamos a poner un ejemplo sobre el que se verá cuál es el proces básico.

Primero, supongamos que la historia ha generado un texto determinado. Se le pide entonces a un conjunto de lectores que evalúen la coherencia del texto, y no la expresión lingüística textual, que, evidentemente, no es correcta. Para evaluar esto, a cada lector se le da una cuartilla con el texto generado y cuatro preguntas, que son las de la Figura 3.19. El texto ha sido retocado desde la generación original para que sea más legible, pero sin cambiar su estructura ni su relación.

Supongamos, entonces, después de tener n respuestas (10 en este ejemplo que mostramos), que conseguimos la Tabla 3.5 de evaluación

	Coherencia	**Orden**	**Conocimiento**	**Distribución**
Lector 1	7	7	4	10
Lector 2	5	8	3	9
Lector 3	6	6	4	10
Lector 4	5	7	2	10
Lector 5	5	6	3	10
Lector 6	4	8	4	7
Lector 7	6	7	2	10
Lector 8	5	6	1	10
Lector 9	7	8	0	10
Lector 10	2	6	2	10

Tabla 3.5: Tabla que representa evaluación de un texto para diez encuestados.

Por lo tanto, a partir de la tabla anterior los siguientes valores, aplicando la media aritmética. Estos valores los podemos ver en una gráfica en la Figura 3.20.

- **Coherencia** = 5,2
- **Orden** = 6,9
- **Conocimiento** = 2,5

Evalúe la calidad del texto con valores del 1 al 10, teniendo en cuenta que se interpretarán aproximadamente como:

0 Ausente.

2.5 Existe, pero es incorrecto o insuficiente.

5 Es aceptable, pero nadie lo expresa así.

7.5 Hay gente que lo expresa así, pero no es la mejor manera.

10 No lo distingo de cómo lo haría un hombre que supiera escribir con corrección.

El texto es el siguiente:

> *Socrates: Does Meno see that the boy has advanced? The boy didn't know before the side of the figure. The boy doesn't know now the side of the figure. The boy have difficulty now, but the boy had not difficulty before.*

Responda a las preguntas según las instrucciones recibidas antes, desde el punto de vista de la estructura lógica del texto, no la representación textual. Tenga en cuenta que es un fragmento de un texto y los personajes ya habrían sido presentados:

1. ¿Cuál es la coherencia entre las frases, existe sentido como unidad?
2. ¿Existe un orden secuencial apropiado entre las frases?
3. ¿Hay un conocimiento de los personajes de la historia adecuado?
4. ¿Existe corrección y buena distribución entre las partes dialogadas y narradas?

Figura 3.19: Cuestionario para la evaluación de historias.

- **Distribución** = 9,6

Y con estos valores ya podemos ajustar la función heurística de la *calidad* del PLAN DE DISCURSO que ha dado lugar a este texto.

$$h(discurso_1) = \frac{5{,}2 + 6{,}9 + 2{,}5 + 9{,}6}{4} = 6{,}05 \tag{3.2}$$

Por supuesto, la evaluación no es perfecta. Muy probablemente ni siquiera el lector entienda perfectamente lo que quiere decir coherencia de la misma manera que lo entendemos desde el punto de vista de la aplicación,

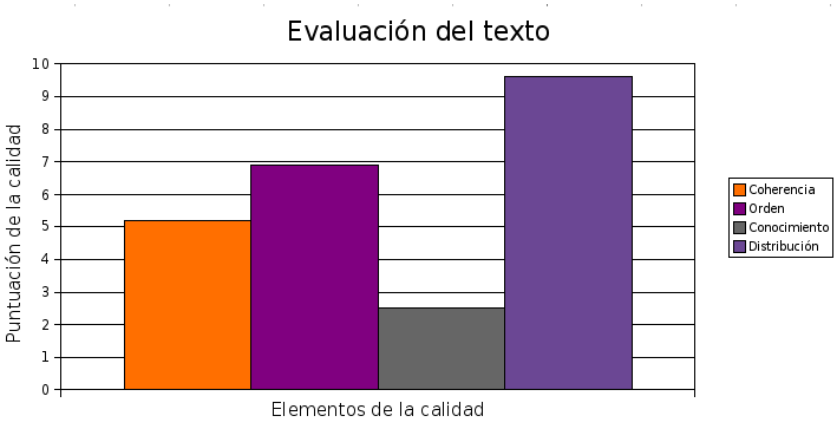

Figura 3.20: Gráfica que representa los valores de la heurística según la evaluación.

por lo que su puntuación nunca será exacta. Además, dar una puntuación no sólo es subjetivo, sino que seguramente el lector no sabrá muy bien la diferencia entre 3 y 4, en una escala del 1 al 10. También hay que saber que ningún lector, a priori, invertirá mucho esfuerzo en valorar realmente la historia, y generalmente no se puede obligar a alguien que lo haga.

Por estas razones sabemos que no tenemos la información más perfecta, pero lo consideramos como una aproximación mínima que nos habilita para seguir perfeccionando la evaluación y estudiando la repercusión de esta en la calidad del sistema final como se ve desde el punto de vista del creador de la misma: lo cercana que es a la manera de escribir textos que tenemos los humanos.

3.3. Salida del sistema: El texto generado

Hasta esta sección hemos presentado el funcionamiento interno del sistema de generación en cuanto a la planificación de contenido se refiere. No obstante, pese a estar en esta parte la aportación principal del trabajo, se ha acometido además la tarea de realizar lo que se conoce en la literatura como Surface Realization, o paso de la representación estructurada y formal de la información sobre una historia en un ordenador a un texto, ya en un idioma natural (español, inglés...) para la posterior legibilidad por un humano.

3.3.1. Traducción de discurso a texto

La propuesta que aquí realizamos consiste en una manera simple pero en ocasiones eficaz que consiste en la traducción directa de los Mensajes que se han producido durante el proceso de generación texto, siendo la traducción

de cada mensaje llevada a cabo mediante la aplicación de una plantilla de texto dependiente de dicho mensaje. Es decir, el mensaje de "conocer algo" tiene una plantilla del estilo de (en inglés):

%SUJETO % knows %DIRECTO % [at %TIEMPO %].

En esta plantilla, sólo habría que hacer las sustituciones, y sólo añadir aquellas partes de las que realmente tenemos información, es decir, si no tenemos la información sobre cuándo ocurrió el hecho en la base de conocimientos, en la plantilla deberíamos omitir el "[at %TIEMPO %]". Así, podríamos tener un mensaje como el de la Tabla 3.6.

Mensaje de Conocimiento	
SUJETO	socrates
DIRECTO	virtue

Tabla 3.6: Ejemplo de un mensaje que va a ser traducido por texto.

Este mensaje podría ser traducido como:

Socrates knows virtue.

De esta forma podemos traducir los mensajes. No obstante, si en la historia realizamos una traducción tan simple, el resultado final va a ser muy pobre. Necesitamos un sistema de traducción que esté por encima de los simples mensajes, porque por un lado tenemos que representar la estructura del texto, y por otro, además, las relaciones entre los diferentes hechos que narramos.

Supongamos que tenemos, pues, una estructura más compleja, todo un párrafo, como elemento de plan de discurso. La Figura 3.21 muestra este párrafo. Vemos que tenemos una relación de *condición* entre un núcleo y un satélite, y ha de ser representado de tal forma que se vea claramente la relación.

```
[PARAGRAPH:
  [NS:
     [CONDITION
       [Meno doesn't know].
     ]
     [Meno asks].
  ]
]
```

Figura 3.21: Ejemplo de párrafo que va a ser pasado a texto.

Para conseguir esto, usamos plantillas de la misma manera que en los MENSAJES, pero ahora las aplicamos a todas las UNIDADES SEMÁNTICAS. Así, los PÁRRAFOS también pueden ser traducidos. En el ejemplo que estamos siguiendo, podríamos tener la plantilla de:

%NUCLEO % %traduccion(RELACIÓN) % %SATELITES %.

Esta plantilla indica que primero hay que escribir el núcleo de la relación, después una representación textual de la relación retórica que uno el núcleo con los satélites, y, después, el conjunto de satélites, que a su vez puede ser traducido mediante la aplicación de otras plantillas diferentes que correspondan según el caso. La traducción de la "RELACIÓN" de la que hablamos consiste en la aplicación de una tabla que equipara las relaciones retóricas con representaciones textuales, de la forma:

Tabla de traducción de relaciones retóricas	
CONDITION	", because "
CONJUNCTION	", and "
...	...

Tabla 3.7: Fragmento de la tabla de traducción a texto de las relaciones retóricas.

Así, siguiendo la Figura 3.21, podríamos establecer la traducción como:

Meno asks, because Meno doesn't know.

Como se ve, con estas simples reglas de plantillas, por lo tanto, ya adquirimos la habilidad de traducir planes complejos a un modo legible. Por supuesto que no es la mejor manera, pero al menos tenemos la posibilidad de disponer de un sistema, gracias a estas ideas, que reciba datos y genere texto, de manera global y desacoplada, ya que es perfectamente posible sustituir, por ejemplo, este realizador superficial que presentamos.

No obstante, aún podemos mejorar un poco más la salida, con ideas simples pero potentes, usando recursos que ya hemos presentado con anterioridad en este trabajo. Lo vemos en la siguiente sección.

3.3.2. La percepción del lector en la traducción

En la Sección 3.2.3 se han expuesto algunas ideas que se han utilizado para modelar la percepción que el lector tiene de la historia que se está creando durante la ejecución del algoritmo de generación de historias en lenguaje natural. Hemos comprobado que es una herramienta útil, ya que nos hace posible guiar la generación de historias según lo que sepa el lector.

También la podemos usar para generar el texto final. Si, durante la traducción del contenido estructurado en formato de datos de ordenador propagamos una estructura que contenga la percepción del lector, y la vamos actualizando según se "escriban" datos en el texto final, podremos saber qué piensa el lector sobre la historia en cada punto de su lectura, con lo que tendremos la información que necesitamos para añadir a la historia textual también pronombres y referencias concretas.

Por ejemplo, si en un punto de la historia no sabemos el nombre de un personaje porque aún no se ha escrito, podemos hacer referencia a él como:

That person spoke about the virtue.

Así podríamos seguir hasta que hubiese un mensaje que consistiese en poner explícito el nombre de ese personaje, (por ejemplo, Menón) entonces podríamos ya hacer referencia a él como:

Meno spoke about the virtue.

Con esta y otras reglas que están incluidas en las plantillas de traducción de los MENSAJES y UNIDADES SEMÁNTICAS en general es posible ya realizar una traducción que permita al lector leer el texto sin problemas, si bien es verdad que la calidad literaria va a ser pobre.

3.3.3. Diferencia entre diálogos y narraciones

El único asunto que queda por exponer en lo que a la realización superficial en nuestro proyecto de investigación se refiere es el relacionado con la diferencia de escritura de los diálogos y las partes narradas. La representación textual relacionada, tal como la escribimos usualmente en español, francés o inglés es distinta: un párrafo por elocución, precedido de un guión y separado de la parte narrativa, por ejemplo. Por tanto, hay que tener en cuenta estos detalles.

Por tanto, vamos a tener un conjunto de mensajes cuya traducción, pese a ser parecida a la de las partes narrativas, tiene una estructura diferente que sigue los patrones de la representación de diálogos a la que estamos acostumbrados. Estos mensajes, por tanto, tendrán claramente diferenciado quién es el autor de la elocución, cosa fundamental, aunque imponemos ciertas restricciones por simplicidad.

Además, hemos de tener en cuenta que durante un diálogo existe una posición de cada personaje con respecto a los demás, es decir, cuando alguien "habla", "habla" con otro personaje, y esto ha de verse reflejado en el texto. Por ejemplo, no es válido generar un texto con la forma de:

Socrates says: Do Meno know virtue?

Cuando en realidad Sócrates está dirigiéndose a Menón. La representación correcta sería:

Socrates says: Meno, do you know virtue?

Así sí, realmente, estaríamos transmitiendo la información correcta. Tenemos la información de a quién se dirige cada personaje en los hechos de la base de conocimiento, por lo que simplemente podemos usarla. Por esta razón y las antes comentadas diferenciamos ambas realizaciones superficiales.

3.4. Ejemplo completo

A continuación, en esta sección, ofrecemos la salida del sistema de un ejemplo concreto de funcionamiento. Para realizarlo se ha creado a mano una descripción estructural, en el formato propio que se ha expuesto anteriormente, de un fragmento del diálogo de Menón escrito por Platón. El texto coincide con algunas frases del original. Se verá que no sólo se consideran las partes de diálogo que aparecen en el original, sino que también se han añadido fragmentos de narración.

Es fundamental tener en cuenta, a la hora de leer el siguiente texto, que la aplicació de objetivos, reglas y plantillas no es, en el prototipo implementado, aún perfecta, y pueden verse, por ejemplo, construcciones sintácticas poco usuales en inglés, o incluso incorrectas. Como trabajo futuro se contempla, por supuesto, la mejora de las plantillas de generación, con lo que la salida será más correcta, y la conexión del PLANIFICADOR DE DISCURSO con un realizador superficial externo que no funciones con plantillas.

En el Apéndice C se muestra cuál ha sido la descripción con el formato de entrada propio de la aplicación de la que se ha generado este texto.

Meno thought that Socrates know Virtue always. Meno asked Socrates:

- Can you tell me: People acquired Virtue because People learned Virtue, or People acquired Virtue because People practiced Virtue?

Then Socrates said to Meno:

- It's false that I knew Virtue.

Then Meno asked Socrates:

- Must I told in Thessaly that you said that It's false that Socrates knew Virtue?

Then Socrates said to Meno:

- Yes.

Then Meno asked Socrates:

- Did you met Gorgias always in Athens?

Then Socrates said to Meno:

- Yes.

Then Meno said to Socrates:

- I thought Idea of virtue, because Gorgias thought Idea of virtue.

Then Socrates said to Meno:

- You told to me Idea-of-virtue.

Then Meno said to Socrates:

- Virtue was Multiple.

Then Socrates said to Meno:

- You said Virtue was Multiple was not coherent with the idea of Bees is Unique always.

Meno was confused. Then Meno said to Socrates:

- Virtue was Courage, and Temperance, and Wisdom, and Magnanimity.

Then Socrates said to Meno:

- God give Virtue to People.

Then Meno said to Socrates:

- I agreed with you.

Capítulo 4

Discusión y propuestas

Hasta este capítulo se ha descrito con detalle cuál es el estado del arte a partir del cual ha surgido esta investigación, cuáles son los principales inconvenientes o aspectos sobre los que merece la pena invertir esfuerzo con el objetivo de aportar nuevas ideas y técnicas, y un sistema que pretende ofrecer nuevas alternativas a algunos problemas en la generación de historias.

Esta capítulo está dedicado a poner de manifiesto, con detalle, y discutir, las principales aportaciones que ofrece la investigación realizada y el prototipo implementado.

4.1. Principales aportaciones

Cualquier sistema que se desarrolle dentro del ámbito de la investigación tiene como objetivo principal el de aportar cualquier creación de conocimiento que se haya originado de su desarrollo sobre el estado de arte en el campo en el que se estudie. De la investigación que se presenta, de este modo, pueden extraerse ciertas ideas concretas que suponen algunas perspectivas, soluciones o conceptos nuevos que merece la pena destacar como aportaciones. En esta sección las comentamos, aislándolas del resto del sistema para hacerlas claras, pero relacionándolas en general con toda la investigación y, sobre todo, con el estado actual de las investigaciones en las que se ha basado este trabajo.

4.1.1. Historias con varios personajes

El sistema nació como intento de crear una nueva solución al problema de las historias que no sólo se dedican a narrar en tercera persona hechos o a transmitir información, sino a contar la vida o acciones de personajes en conjunto de tamaño no limitado de ellos. Desde luego, no es el primero de ellos, como podemos comprobar en la Sección 2.2.

Sin embargo sí hemos contemplado un conjunto de características que

hemos considerado fundamentales para la investigación y que suponen una aportación en el campo de la generación de historias en lenguaje natural. Nuestro sistema de generación tiene la capacidad de crear narraciones a partir de un número indeterminado de personajes, un número que es potencialmente muy grande (cientos de personajes). Esta característica se diseñó porque disponíamos de unos sistemas de agentes que podían generar su salida como *log* de datos, y estas grabaciones de los mismos no eran, en absoluto, legibles para un ser humano.

Por tanto, disponer de un sistema, ya sea de una calidad literaria excelente o no tan buena, que permite realizar una traducción de varios miles de datos numéricos u ordenados como grandes tablas a una representación textual que se comporte como un sumario de esta información tiene un interés directo como ayuda al trabajo humano, si tiene que depurar esos datos, o, quizás, como generación de informes, o como resumen de una partida de muchos jugadores, tras la que narramos la intervención del jugador que más puntos ha ganado, por ejemplo.

4.1.2. Narración y diálogo mezclados

No existen sistemas significativos capaces de generar planificación del discurso y realización superficial con diálogos y narraciones mezcladas. Ésta es la aportación principal de nuestro trabajo. Como hemos comentado en el Capítulo 2, los trabajos relacionados con los problemas que afronta esta investigación no trabajan considerando el diálogo.

Nuestro modelo de diálogo no es, desde luego, perfecto. Sin embargo, sí que hemos creado una distinción clara entre los eventos de narración o acción y los de la comunicación entre distintos personajes. A pesar de que se han tomado muchas soluciones simples, conseguimos que el sistema se comporte de manera diferente según el tipo de historia que queremos narrar, lo cual ofrece un tipo de generación de historias que puede dar algunas ideas sobre cómo tratar o contar pasajes de historias mezclando partes dialogadas con narradas, lo cual, pese a que desde el punto de vista del autor puede suponer un esfuerzo, y no facilitar la tarea de contar historias, para el lector puede significar la diferencia entre una lectura interesante o una aburrida, una didáctica o una que lo deje indiferente.

4.1.3. Percepción de la historia por parte del lector

Muchos planificadores de discurso usan modelos del lector. De hecho, podemos decir que siempre existe, aunque sea implícito, un modelo de lo que el lector sabe o quiere recibir. Este modelo es más que necesario, ya que sin él no se sabría qué hay que escribir en una historia, ni siquiera a mano. Por tanto, la explicitación del modelo en sí no es una aportación fundamental.

Sí lo es, sin embargo, la inclusión del modelo BDI de Bratman en la percepción del lector de la manera en la que lo proponemos. Esta inclusión se ha debido al carácter "multiagente" de nuestra aplicación. Necesitamos las características que nos proporciona este modelo porque, como tenemos que narrar lo que les ocurre a varios personajes, es "obligatorio" explicar sus *deseos*, *creencias* e *intenciones*, del mismo modo que usarlas para guiar la creación de la historia.

Al usar un modelo BDI en la planificación podemos utilizar los mismos mecanismos que se usaron durante la generación de la historia para la narración de la misma. De este modo, tenemos información que nos permite crear discursos e historias con un contenido más cercano a la ejecución real de los hechos y, por tanto, más interesante para el lector. En [YM94] podemos encontrar trabajos relacionados con esta aproximación, donde se presenta un algoritmo que pretende enfocarse en los personajes, extrayendo las posibles motivaciones que los condujeron a realizar las acciones que realizaron.

4.1.4. Sistema completo de generación de historias

Otra de las principales ventajas del proyecto de investigación que se presenta en esta memoria es la de ofrecer un sistema completo de generación de historias en lenguaje natural. Muchos de los sistemas que se han comentado durante la exposición del trabajo previo simplemente estudian casos concretos o se limitan al trabajo sobre operaciones muy determinadas de la generación de historias.

Desde luego, esas propiedades de los trabajos no disminuyen su calidad como tales. Sin embargo, éste que se presenta se incluye dentro del grupo de aquellos que, genera historias completas ya de manera textual, si bien es verdad que algunas fases se solucionan de una manera simple, y que es necesario que se avance en su desarrollo.

4.2. Aplicaciones del generador de historias

Hemos avanzado, hasta aquí, que el sistema que se ha presentado es capaz de manejar datos provenientes de fuentes de programas de ordenador capaces de generar datos que representan la vida o ejecución de un colectivo de personajes con mayor o menor complejidad o autonomía. Ahora exponemos algunos de estos sistemas, con los que ha sido desarrollada y probada la aplicación.

Las narraciones que se generan en este proyecto están fundadas en un conjunto de datos que pueden provenir de diferentes fuentes. Se requiere, por la orientación del programa creado, que estas fuentes estén formadas por colectivos de unidades individuales que sean capaces de actuar, en el sistema adecuado, de manera independiente de las demás, de forma que cada unidad pueda grabar una traza de operación.

Tras la ejecución de una actividad dada, se recoge la traza de cada una de las unidades que ha desempeñado un papel en dicha actividad, y se exporta, como comentaremos más adelante, para que un programa de creación de texto en lenguaje natural pueda generar una historia que narre los sucesos importantes, de la mejor manera posible.

El proceso general que aplicamos durante esta parte del proyecto es la de añadir información a los datos de entrada. En cada paso, mediante datos adicionales, reglas o algoritmos, añadimos contenido explícito a la cadena de entrada del sistema generador de forma que, cuando termine el proceso de generación de contenido, el realizador de texto, encargado de dar la forma final textual, sea capaz de escribir una narración.

4.2.1. Barcos autónomos cooperantes a escala

En ciertas operaciones reales, la seguridad del hombre puede verse comprometida durante su realización. Por ejemplo, en un escenario de recogida de una mancha de petróleo en el mar, las emisiones contaminantes pueden perjudicar a los tripulantes de los barcos. En algunas situaciones de rescate de náufragos, puede darse la casualidad de que salvar a una persona ponga en peligro la vida de todo un conjunto de rescate. En el marco de todo este tipo de operaciones se ha desarrollado, como proyecto paralelo y enlazado con este trabajo de investigación, una plataforma de robots autónomos marinos que son capaces de realizar de manera no tripulada una serie de operaciones en las que es decisivo, por una parte, eliminar la presencia del ser humano (o sin la necesidad, puede ser recomendable), y, por otro, la labor de modo cooperativo, ya que una sola unidad robótica no es capaz de completar la misión con éxito, por las características de la misma.

A partir de este sistema se producen trazas susceptibles de convertirse en textos es un entorno de experimentación sobre maniobras marítimas de vehículos acuáticos no tripulados. Para la experimentación real, estos vehículos son emulados con robots autónomos montados a partir de barcos de radio control. Es posible encontrar más información en [LGSE06, GSDL06]. Los barcos son capaces de efectuar maniobras de grupo automáticamente. Están orientados al estudio a escala de escenarios reales como la recogida de una mancha de petróleo, la salida de un grupo de naves de un puerto, o el rescate de náufragos. Están programados para realizar tareas de un modo cooperativo, de modo que, dado un objetivo claro, puedan encontrar la mejor solución entre ellos y operar de modo colectivo para llevarla a cabo. En la Figura 4.1 se muestra una fotos de los barcos robóticos.

Estos barcos han sido equipados con una unidad electrónica de control y cálculo, empaquetada en lo que ha dado en llamarse *Caja Universal*. Cada una de estas unidades puede ser desacoplada del barco, y utilizada en cualquier otra unidad que cumpla un conjunto simple de requisitos. De este modo, se ha construido un sistema de manejo simple de robots útil para un

Figura 4.1: Foto de los barcos autónomos a escala.

amplio grupo de sistemas. En la Figura 4.2 puede verse una fotografía de la caja con la electrónica, y un esquema de su funcionamiento en la Figura 4.3. Las funciones que ofrece esta caja de control inteligente automático son las siguientes:

- **Comunicación**, para que los robots autónomos o barcos puedan intercambiar información sobre el proceso y el estado de la operación, entre ellos y el operario que esté supervisando la operación.
- **Gobierno de los motores del barco**, gestión de los sensores del barco y control de rumbo y posición. Los barcos son capaces de maniobrar en un entorno marino de forma autónoma, y, por tanto, la electrónica de a borde debe proveer estos servicios a los algoritmos de alto nivel, con operaciones de tipo *ir a un lugar* o *mantener rumbo*.
- **Gestión de datos**. Durante el desarrollo de una operación, surge una cantidad muy grande de datos que son susceptibles de almacenar y estudiar, para revisiones y control de las capacidades y estados de las decisiones y comunicación de los barcos.
- **Inteligencia artificial**. La aportación básica del sistema es, evidentemente, una Inteligencia Artificial capaz de resolver problemas concretos relacionados con el entorno en el que se plantea el problema. Los programas de a bordo contienen algoritmos de Inteligencia Artificial

capaces de solucionar situaciones en las que la cooperación es necesaria para llegar a alcanzar el objetivo dado. Toda la Inteligencia Artificial de los barcos, para poder ser realmente operativa, se apoya en las tres funciones anteriores de *comunicación*, *control* y *gestión de datos*.

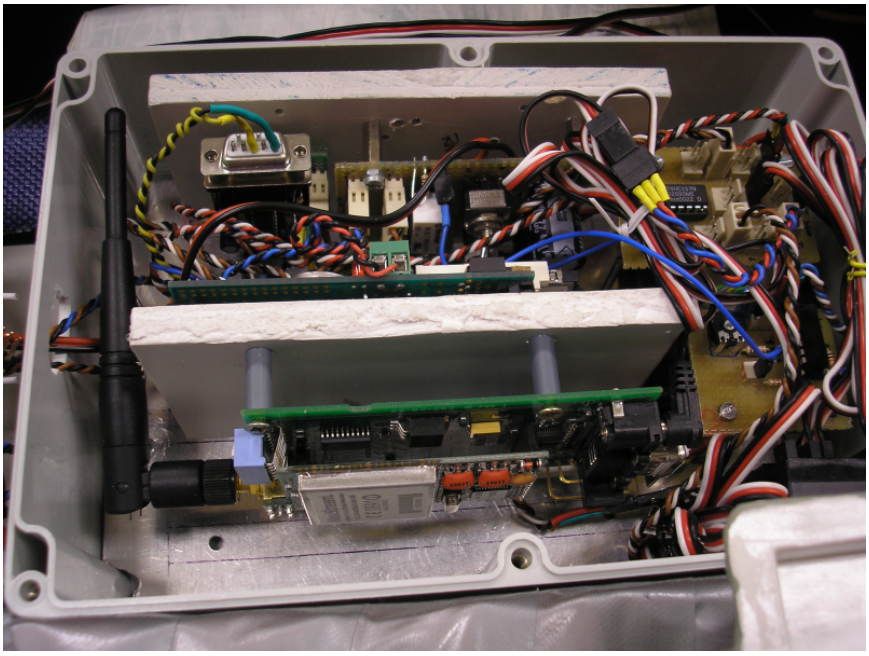

Figura 4.2: Foto de los caja de control de los barcos.

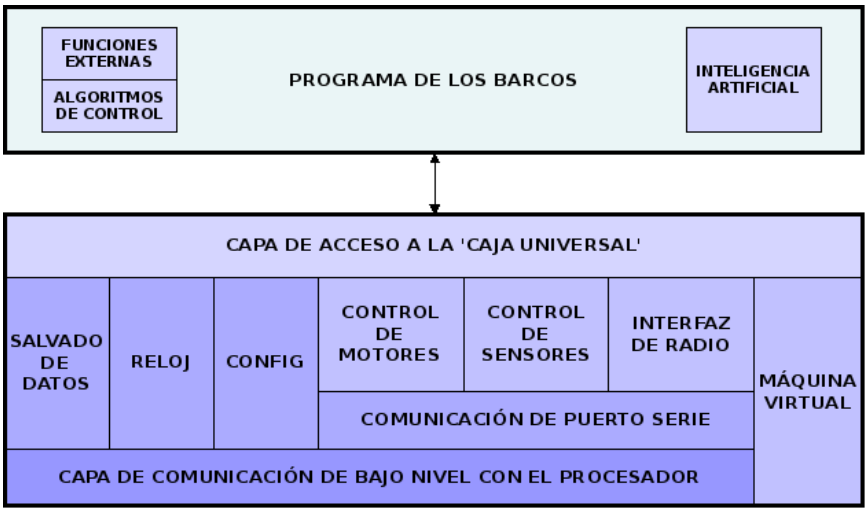

Figura 4.3: Esquema del programa de control de la caja.

Todas estas funciones, consideradas como una capacidad global, dotan a los barcos de propiedades útiles para la cooperación en la resolución de problemas experimentales reales.

A continuación mostramos la historia generada a partir de una ejecución en la plataforma de experimentación simulada de los barcos, en la que dos de ellos están enlazados por una red, y tienen que realizar una maniobra de giro complicada. En este tipo de maniobras es necesaria la cooperación, que aquí representamos como un diálogo. En el Apéndice D se muestra el archivo de entrada correspondiente a este texto. Hay que tener en cuenta que existen errores ortográficos y gramaticales porque las plantillas tienen una capacidad limitada de generación.

> *Ship 1 was located in Location a, and Ship 2 was located in Location b. Then Ship 1 said to Ship 2:*
>
> *- I received Message.*
>
> *Then Ship-1 said to Ship 2, because Ship 1 received Message, and because Ship 1 have attached Net always, and because Ship 2 have attached Net always:*
>
> *- I turn to Location c.*
>
> *Then Ship 2 asked Ship 1:*
>
> *- Do You knew Method?*
>
> *Then Ship 1 said to Ship 2, because Ship 1 knew Method:*
>
> *- Method was Following always.*
>
> *Then Ship 2 said to Ship 1:*
>
> *- Ok.*
>
> *Then because Ship 1 received Message Ship 1 moved to Location c, and Ship 2 moved to Location d.*
>
> *Then Ship-1 controlled Direction.*
>
> *Then because Ship 1 received Message Ship 1 stopped, and Ship 2 stopped.*

4.2.2. Simulaciones de agentes sociales

Otro de los sistemas que ha sido usado como fuente de datos para el generador de texto consiste en una aplicación informática que modela y simula el comportamiento de sociedades de personas que son capaces de crear diferentes tipos y grados de relaciones entre sí [PAHS06].

El programa simula el comportamiento de una sociedad durante una serie de años, dadas unas condiciones iniciales determinadas. En esta simulación, cada entidad social (la representación simulada de una persona) es capaz

de relacionarse con aquellas otras entidades cuya "cercanía" a la primera no exceda un límite. Esta cercanía no representa, necesariamente, cercanía local en la simulación (es decir, no quiere decir que dos personas vivan juntas), sino cercanía relacional, o cómo de probable es que se forje una relación.

Las entidades sociales de la simulación evolucionan durante la misma. Cuando se han producido relaciones entre dos de estas entidades, surgen unos cambios en éstas ocasionados por la diferencia que posiblemente haya entre sus características. Por ejemplo, es posible que una persona con convicciones religiosas fuertes influya a otra con unas convicciones del mismo tipo, pero más débiles.

Las relaciones entre las personas simuladas, en el caso de que se cumplan unas determinadas reglas, pueden dar lugar a uniones entre dos personas, que producen hijos. Estas reglas, evidentemente, son del tipo de tener diferente sexo, ideologías y religiones parecidas, y otras. Así, las poblaciones evolucionan, no sólo por medio del cambio de estado de sus individuos, sino también como consecuencia de su crecimiento de población absoluta. Esto, claro, puede conllevar nuevas relaciones, y, por lo tanto, nuevas tendencias sociales.

El conjunto de parámetros que definen el estado de la sociedad en el momento de inicio de la simulación pueden ser establecidos por el usuario del sistema, de modo que existe la posibilidad de crear diferentes tipos de experimentos simulados, con resultados potencialmente diferentes.

Con este entorno se ha enlazado el generador de historias, y se han creado narraciones de nivel cualitativo de los agentes que interaccionan en este medio. Estas narraciones son puramente descriptivas, y trazan la vida de cualquiera de estos personajes. A continuación explicamos cómo se ha realizado el proceso de generación de historias a partir de este sistema. En la Figura 4.4 mostramos la aplicación de simulación social.

Adaptación del sistema multiagente a un dominio fantástico: emulación de Juegos Multijugador

El reciente éxito de los videojuegos de rol multijugador *on-line* masivos (conocidos como de MMORPGs[1]) ha hecho que crezca el interés en su desarrollo y en las posibilidades de crecimiento que tienen, tanto desde el punto de vista de la innovación tecnológica, como de producción y beneficio. En estos juegos, una gran cantidad de personas (en ocasiones, cientos de miles), conviven virtualmente en un entorno en el que puede relacionarse entre ellos, y con otros personajes no controlados directamente por humanos, sino mediante Inteligencia Artificial.

Según este funcionamiento, el transcurso de una partida (que puede durar meses) provoca que se forjen relaciones entre estos personajes. Cuando

[1] *Massive Multiplayer Online Role Playing Games*

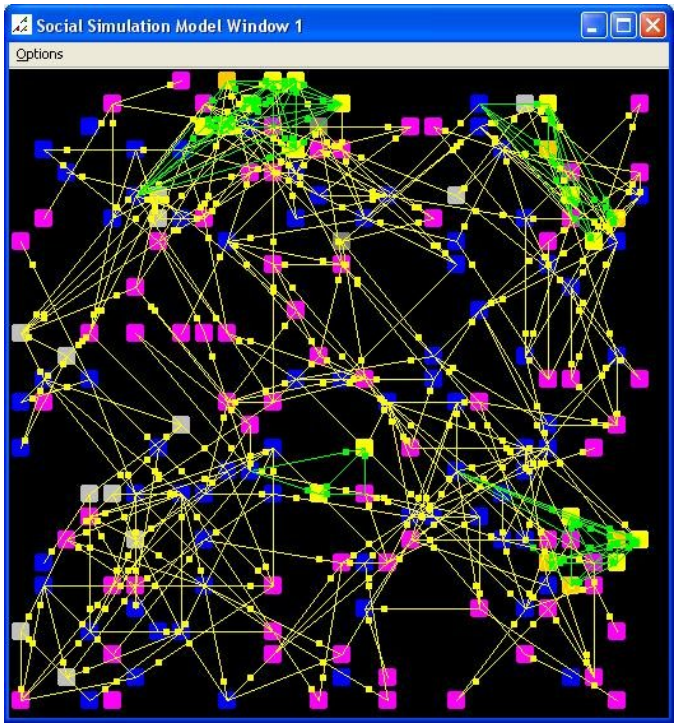

Figura 4.4: Aplicación de simulación social multiagente.

un humano interacciona con un un *Personaje No Jugador*[2], es posible que necesite, o le sea apropiado, conocer la historia pasada de este personaje, ya sea para la consecución de una tarea en el juego, o para mejorar la calidad del mismo. En la Figura 4.5 se puede ver una captura de pantalla de un juego multijugador por Internet.

Es posible que, dentro de la arquitectura de un juego de este tipo, se disponga de capacidades de almacenamiento de trazas de relaciones y comportamientos. Estas trazas, generalmente, quedarán almacenadas en una estructura de datos que, posiblemente, esté dotada de un orden lineal por el tiempo en el que se han sucedido los hechos. Sin embargo, al jugador le es de muy poca utilidad, dentro del entorno del juego, recibir una traza que represente la historia del personaje en este formato. Por esto, el personaje controlado por Inteligencia Artificial debería ser capaz de narrar su historia de una manera textual.

Por otro lado, la generación de narraciones a partir de trazas de personajes no sólo tiene la utilidad de la interacción con el jugador, sino también como método de información sobre el transcurso de una partida para un público que no esté tomando partido en ella, ya sea el administrador del

[2]Este término es más conocido en la literatura como *Non-Player Character* (NPC), en inglés.

Figura 4.5: Captura de pantalla de un juego multijugador masivo por Internet.

sistema, que tendría la posibilidad de leer de una forma textual cuáles han sido los hechos más reseñables en el juego, o usuarios que deseen conocer qué historia ha surgido durante una partida, narrada de un modo atractivo para ellos, y quizá publicada en una página web.

Por estos motivos, se decidió adaptar el sistema multiagente comentado antes de modo que pudiésemos disponer de una salida similar a la de un juego multijugador.

Para emular historias fantásticas, cuya narración tiene, en general, mayor atractivo para un lector, se llevó a cabo una migración del dominio del sistema de agentes sociales [LHG07]. Con este cambio, se dotó al sistema de un dominio semántico muy diferente al de la pura simulación social, y en la historia ahora podían aparecer personajes fantásticos como elfos, enanos y orcos. Con estos personajes, y nuevos atributos que no son simplemente sociales se hizo posible la generación de un contenido en un dominio mucho más atractivo, desde el punto de vista del entretenimiento, y las historias pudieron ser más ricas en contenidos.

Las relaciones posibles que podían crearse entre los agentes sociales (convertidos ahora en personajes fantásticos de fábula) fueron modificadas, y aparecieron *traiciones*, *romances*, *enemistades*, y demás. Así era posible la creación de un grupo social cuya vida estuviese compuesta por sucesos variados y algo más emocionantes.

Además, se añadió al sistema la posibilidad de generar eventos aleatorios que, junto con las relaciones sociales más emocionantes, dotasen ya a la historia de un conjunto posible de datos de entrada mucho más rico. Estos eventos aleatorios consistían en, por ejemplo, "matar a un dragón", "encontrar un tesoro escondido" o "librar una gran batalla". De este modo ya las historias posibles se hacían mucho más numerosas, y las reglas de generación de historias podían crean contenido más entretenidos y creativos.

Heródoto

Desde luego, este trabajo de investigación ha sufrido una gran evolución desde su inicio. La primera versión que se publicó de él [LHG07, HPML07, HLGH07] fue llamada Heródoto[3]. A partir de las sucesivas ampliaciones y mejoras de los prototipos surgió la versión final que ahora presentamos. Sin embargo, tiene interés mostrar los resultados que se consiguieron con las primeras versiones, y de qué modo se obtuvieron.

En Heródoto la planificación del contenido no se realizaba mediante búsquedas ni esquemas, sino aplicando plantillas más o menos complejas con las que era posible narrar. Heródoto, además, estaba muy centrado en las operaciones de Determinación del contenido. El programa recibía un conjunto muy grande de hechos provenientes de un sistema multiagente de simulación social (explicado en la Sección 4.2.2), y la tarea de Heródoto consistía en filtrar la información no relevante del conjunto de datos de entrada, y elegir, mediante un conjunto de reglas y fórmulas matemáticas que orientaban la decisión, los hechos que debían narrarse. Finalmente, el programa era capaz de generar una salida textual que era un resumen de las historias más relevantes que se contaban.

Determinación del contenido en Heródoto

Heródoto estaba centrado en la determinación del contenido desde una cantidad ingente de datos, y la presentación ordenada de los mismos. Por tanto, conseguir una determinación del contenido potente era fundamental. Para esto se crearon un conjunto de valores numéricos que se asignaban a los hechos según una serie de funciones y reglas que permitían valorarlos, o establecer un *interés* en ellos que crease una clasificación de qué debía y qué no debía ser contado. Este interés estaba dividido en dos valores: el Interés base y el Interés de las relaciones- El Interés base se calculaba con la Fórmula 4.1:

$$I_b(X) = \sum_{i=1}^{n} f_i \cdot h\,(X, i) \tag{4.1}$$

[3]Heródoto (Halicarnaso 484 adC – Atenas, 425 adC) es considerado el padre de la historiografía.

Básicamente, este valor se calculaba buscando, para cada hecho del personaje, en una tabla, el valor de dicho hecho (f_i), y ponderándolo con las características del personaje ($h\,(X, i)$), haciendo que, por ejemplo, casarse entre dos cristianos, o dos elfos (según el dominio, HERÓDOTO se aplicó a varios) tuviera menos interés que un matrimonio entre un cristiano y un musulmán, o un elfo y un orco. El sumatorio de cada uno de los hechos de un personaje según esta fórmula nos permitía calcular su INTERÉS BASE.

El INTERÉS DE LAS RELACIONES, como podemos ver en la Fórmula 4.2, consistía en encontrar el *interés* que tenía un personaje dado por las relaciones que tenía con otros personajes, haciendo que aquellos "amigos" de personajes famosos fuesen más interesantes ($I_b(Y)$). Del mismo modo que el INTERÉS BASE, era ponderable por las características de ambos personajes (X e Y), según el término $g\,(X, Y, i)$.

$$I_r(X) = \sum_{i=1}^{n} I_b(Y) \cdot g\,(X, Y, i) \tag{4.2}$$

Finalmente, a cada hecho se le asignaba un interés que se computaba con la suma de ambos valores antes calculados, como mostramos en la Figura 4.3:

$$I_f = I_b + I_r \tag{4.3}$$

Con este valor y un conjunto de reglas para aplicarlo, como eliminar todos aquellos hechos redundantes o los que tuvieran un interés que estuviera por debajo de un umbral establecido a mano, creábamos un subconjunto de hechos que estábamos preparados para ser ordenados y traducidos a texto.

Comparado con nuestro sistema, HERÓDOTO sí tenía una determinación de contenido explícita. En la aportación que está presentada en este documento, como se ha comentado, no aplicamos operaciones de este tipo porque la determinación del contenido que hacemos está basada en simplemente no usar el conocimiento de la base de datos de entrada, con lo cual, más que evidentemente, esos datos quedan omitidos. Son, por supuesto, dos alternativas diferentes, con resultados, ventajas e inconvenientes diferentes. Aún así, ambas pueden ser válidas para las mismas aplicaciones. Por ejemplo, podríamos filtrar, aplicando estas ideas, los datos de entrada que tenemos, y después aplicar la planificación del discurso, o, en HERÓDOTO, omitir estas fórmulas y aplicar la búsqueda en las plantillas de la determinación de contenido que explicamos en la sección siguiente.

Planificación del discurso en Heródoto

La planificación del contenido se realizaba de manera muy diferente a como se hace en esta investigación. Se usó un conjunto de plantillas fijas de narración, en las que sólo era posible establecer un orden en el que se

narraban los hechos más importantes de la vida de un personaje junto con sus relaciones con otros personajes o agentes con los que había existido una relación durante la simulación social multiagente.

Con este método no era posible, pues, tener un plan de discurso flexible. Lo único de lo que era capaz el sistema era de crear resúmenes de un personaje principal. Sin embargo, esta característica la hacía interesante desde ciertos puntos de vista, como por ejemplo narrar partidas multijugador, contando los hechos más importantes de cada personaje.

Como ampliación a HERÓDOTO, en [HLGH07] se añadió la capacidad de gestionar un *foco* que podía apuntar a diferentes personajes durante la generación, para narrar la vida de estos, y hacer que la historia fuese más dinámica superando la limitación de un solo personaje por historia. El cambio de foco de un personaje a otro estaba gobernado por un conjunto de reglas que nunca llegó a ser perfeccionado, por lo que el *foco* nuca consiguió un comportamiento respecto al cambio suficientemente bueno.

Desde luego, el sistema de generación presentado en este trabajo es mucho más potente, principalmente en la tarea de planificación del discurso, que es en la que se ha invertido un gran esfuerzo para conseguir una generación de historias rica y suficientemente general.

Realización superficial en Heródoto

Hemos visto hasta aquí que las operaciones de planificación del discurso que aplicamos en HERÓDOTO no son nada similares a las que presentamos en este trabajo. Sin embargo, la realización superficial que se aplica en esta investigación es una ampliación de la que aplicábamos en HERÓDOTO. De igual forma aplicábamos plantillas basadas en los mensajes que formaban el texto final, y también conservábamos una versión primitiva de lo que se convirtió un la percepción de la historia por parte del lector.

Ejemplo de Heródoto

A continuación mostramos tres ejemplos del funcionamiento de HERÓDOTO, según las tres versiones que existen. El primer ejemplo es del trabajo publicado en [LHG07], y vemos que simplemente es el resumen de la vida de un personaje fantástico:

> *The Great Story - A fantasy Middle-Age world:*
>
> *Badash Taltaur the Elf was born in 504.*
>
> *Badash Taltaur met Amdor Taltaur, and she was lost in a forest, then she was enchanted with the incredible spell of memory, then she found a Magic Ring.*
>
> *Badash Taltaur was lost in a labyrinth, then she met Werlom Mcknight, and Werlom Mcknight was offspring of Rirbag Great-*

gibber, and Badash Taltaur was involved in a great battle, then she was enchanted with the incredible spell of frog.

Badash Taltaur fell in love, desesperately, with Werlom Mcknight, then she was lost in a forest, then she found a Treasure, then she married Werlom Mcknight, then she had a child: Idrin Taltaur.

Badash Taltaur had a child: Dora Taltaur, then she had a child: Dwalin Taltaur, then she had a child: Pimmam Taltaur, then she had a child: Baradadan Taltaur, then she found a Magic Sword.

Badash Taltaur found a Magic Ring, then she was lost in a forest, then she was involved in a great battle, then she was enchanted with the incredible spell of sex, then she was lost in a forest.

Badash Taltaur found a Treasure.

Badash Taltaur died in a mysterious accident in 555.

The end.

El segundo ejemplo parte de la versión con *foco* del planificador del discurso de HERÓDOTO, puede encontrarse más información en [HLGH07]:

Jeanine Avery was born in 520.

Jeanine Avery was saved by the priest. Jeanine Avery killed the ogre. Jeanine Avery was involved in the battle. Jeanine Avery was enchanted with the marvellous spell of the frog. Jeanine Avery killed the dragon. Jeanine Avery was lost in the forest. Jeanine Avery met Luisa Brandagamba. Luisa Brandagamba was born in 529. Luisa Brandagamba met Jeanine Avery. Jeanine Avery killed the ogre. Jeanine Avery fell desperately in love with Bobbie Beasttongue. Jeanine Avery inherited the castle. Jeanine Avery met Pogor Brandagamba.

Pogor Brandagamba was born in 529. Pogor Brandagamba killed the ogre. Pogor Brandagamba met Jeanine Avery.

Jeanine Avery grew up. Jeanine Avery fell desperately in love with Bobbie Beasttongue. Jeanine Avery met Haurk Avery.

Haurk Avery was born in 542. Haurk Avery found the magic sword. Haurk Avery met Jeanine Avery.

Jeanine Avery was lost in the labyrinth. Jeanine Avery found the treasure. Jeanine Avery was enchanted with the marvellous spell of the memory. Jeanine Avery was enchanted with the marvellous spell of the frog. Jeanine Avery was involved in the battle. Jeanine Avery killed the ogre. Jeanine Avery killed the dragon. Jeanine Avery was involved in the battle. Jeanine Avery was lost

in the forest. Jeanine Avery found the treasure. Jeanine Avery killed the dragon.

Jeanine Avery was betrayed and killed by Morrain Avery.

Finalmente presentamos el trabajo de la versión de HERÓDOTO que operaba directamente con la simulación social; disponible más información en [HPML07]:

Rosa Pérez was born in 1955, and she met Luis Martínez, and she met Miguel López, and she met María Valdés, and she suffered a horrible childhood, and she had a very good friend: María Valdés, and she believe in God, and she went to church every week, and she met David García, and she wanted to be a priest, and she suffered an incredible accident, and she met Marta Alonso.

When she was a teenager, she messed with a gang, and she met Claudia Sánchez, and she went to confession every week, and she had problems with drugs, and she became an adult, and she met Marci Boyle, and she was involved in a labour union, and she met Carla González, and she got arrested, and she learned how to play the guitar, and she became a hippy, and she was involved in a NGO.

She met Sara Hernández, and she stopped going to church, and she met Marcos Torres, and she fell in love, desperately, with Marcos Torres, but in the end she went out with Miguel López, and she lived together with no wedding with Miguel López, and she had a child: Melvin López. She had a child: Andrea López, and she met Sergio Ruiz, and she separated from Miguel López, and she went out with Sergio Ruiz, and she lived together with no wedding with Sergio Ruiz.

She had a abortion, and she bought a house, and she had a depression, and she had a crisis of values, and she was involved in a NGO, and she had a huge debt, and she inherited a great fortune, and she met Daniel Lorenzo, and she bought a car, and she was unfaithful to Sergio Ruiz with another man, and she was fired from her job.

Nowadays she is an atheist.

4.3. Creatividad del sistema

En la Sección 2.3 se ha mostrado el estado del arte y las ideas principales sobre CREATIVIDAD COMPUTACIONAL. Este concepto se ha incluido en el trabajo con el propósito de motivar una pequeña reflexión sobre las

capacidades creativas que se han logrado con nuestro sistema de creación de historias. En esta sección la desarrollamos.

Obtener un sistema de PLANIFICACIÓN DE CONTENIDO que pueda ser considerado *creativo* no es, evidentemente, una tarea sencilla. Disponer de un sistema así implicaría haber diseñado e implementado un sistema que tuviera la capacidad, vista desde un punto de vista funcional, de "comprender" los datos de entrada como lo hacemos los humanos. Después debería, tras la inteligencia de los datos, ser capaz de crear un objetivo de generación de la historia, es decir, decidir de manera automática qué es lo más susceptible, interesante o pertinente de ser contado. Y, desde luego, crear un hilo de historia tal que presente ciertas características que nos hagan percibir una historia "creativa", haciendo que su contenido sea *original* o *sorprendente*.

Desde luego, esto no es posible hoy en día. Lo que acabamos de comentar es un proyecto demasiado ambicioso dado el conocimiento que hoy se tiene sobre las capacidades de comunicación lingüística del ser humano. No obstante, no conocer algo no implica dejar de avanzar hacia su conocimiento, y este sistema que hemos presentado a lo largo del desarrollo de esta memoria para el trabajo de investigación intenta dar algunos pasos hacia la consecución del objetivo de modelar la generación humana de textos e historias.

Podemos decir que la creatividad de nuestro sistema, por tanto, está almacenada en el algoritmo de generación. Cuanto más "creativo" consigamos que sea este proceso, más "creativa" podrá ser la salida. Como se ha explicado, la inserción de la información que realizamos durante el proceso de generación contiene el conocimiento humano necesario para generar historias. Por ejemplo, los OBJETIVOS están formados por reglas de producción en las que tenemos la posibilidad de "insertar" creatividad. Cuanto más elaboradas sean estas reglas, más elaborada podrá ser la salida, por tener el sistema más posibilidades de creación de estructuras. Las heurísticas que definimos para guiar la búsqueda en el espacio de estado durante la ejecución de la generación en el proceso también incluyen mucha información sobre la calidad de los textos, parte fundamental de lo que conocemos intuitivamente como *calidad creativa*. La representación que establecemos de lo que el lector piensa sobre el estado de la historia también sigue ciertas reglas que almacenan, de nuevo, nuestra creatividad como programadores.

Por tanto, la conclusión a la se puede llegar tras en análisis de estas ideas es que nuestro sistema tiene *tanta creatividad como podamos transmitirle en el modelado del sistema*. Con esta idea, la calidad de la generación de textos depende totalmente de las reglas que hayamos introducido en la generación.

4.4. Comparación de la generación de texto con otros sistemas

Los sistemas de Generación de Lenguaje Natural son capaces de producir un contenido textual de mayor calidad que otros sistemas de generación textual, entendiendo por tal la facilidad de comprensión del texto por parte de un humano, y el grado de similitud del contenido generado con el contenido que generaría otro humano.

4.4.1. Texto y gráficos

En muchas ocasiones, los datos guardados en una ejecución de un programa o sistema de captura de datos dado no sólo son susceptibles de ser narrados, sino también de ser representados de manera gráfica. Por ejemplo, los datos del control automático de un robot, como el muestro de un sensor de posición pueden ser mostrados como una gráfica con el valor de la medida respecto del tiempo, y, para un experto, probablemente esta imagen tenga mucho más valor que un texto descriptivo que comente los datos. Sólo hay que recordar el proverbio antiguo que dice que "una imagen vale más que mil palabras".

No obstante, esto no siempre es cierto. Para empezar, existen muchos sistemas en los que la representación gráfica de la información no es posible, o no es trivial. Por ejemplo, sería muy complicado representar con una imagen el ejemplo con el que hemos hilado la presentación de este trabajo, un diálogo platónico. Una sola imagen no puede capturar, en la mayoría de los casos, hablando en general, todo un discurso de acciones y diálogos. Por eso la aportación de este trabajo tiene interés en el campo de la comunicación hombre-máquina.

Por esto, es importante notar que, en general, el interés de usar una aplicación de Generación de Lenguaje Natural depende del dominio, muy concreto, de aplicación. En esta investigación se propone un dominio de narración en la que los personajes dialogues y realicen acciones, y en una sola imagen es muy difícil capturar toda esa información y transmitirla de un modo claro para el lector.

4.4.2. Texto y voz

Existe una gran diferencia entre los textos escritos y hablados. De hecho el estudio de lenguaje diferencia bien estas dos áreas de la comunicación humana. Ciñéndonos a la narración de historias, oír una historia hablada o leerla en un libro son dos experiencias que pueden ser muy diferentes. Una historia escrita representa con caracteres y esquemas visuales, muchas veces, gran parte de la información que, cuando se habla, se muestra mediante cambios de entonación, o incluso de timbre de voz.

Por ejemplo, en un diálogo, un buen narrador de cuentos adoptará la voz del héroe o del malvado en función de quién esté hablando, y mostrará emoción o miedo según sea necesario. En los textos escritos esto sólo se puede hacer usando caracteres como exclamaciones o interrogaciones que muestren al lector más información que la simplemente almacenada en las palabras. Los textos escritos permiten que el lector interprete, pues, estos "marcadores" del texto como él considere oportuno o le surja, mientras que en las historias oídas el narrador es que establece estos significados.

Sin embargo, podemos comentar en el marco de esta investigación que es posible traducir un texto escrito de manera automática a sonido usando un sintetizador, e incluso es posible hacer que la narración tenga emociones, como queda publicado en [FGGL07]. Un sistema de este tipo recibe un texto marcado con las emociones correspondientes, tarea que se puede realizar de modo automático, y ajustar los parámetros de la prosodia de forma adecuada para la expresión de emociones.

4.4.3. Texto y animación

Los contenidos de animación (incluyendo el sonido) tienen unas capacidades de transmisión de información que superan las del texto escrito y las imágenes fijas. Las animaciones reproducen con mucha fidelidad los entornos en los que se desarrolla la vida humana, por lo que las personas están muy adaptadas a la recepción de este tipo de contenido. Por tanto, comparar la generación de texto con la animación no tiene mucho sentido desde el punto de vista de la oposición, sí quizás viendo la animación como una ampliación de la generación de texto.

Como se ha detallado en todos los capítulos anteriores, el trabajo que se ha realizado está centrado en la Planificación del contenido. Esta parte de la GLN no considera la realización lingüística de las historias generadas, por lo que mucha parte de la ciencia que investiga este tipo de operaciones puede ser aplicada a la generación de historias por animación. Desde luego, habría que traducir la realización de conceptos a texto por la traducción de conceptos de mensajes a movimientos, enfoques de cámara, expresiones y demás. Es, obviamente, un objetivo mucho más complejo y ambicioso, si bien, por otro lado, los resultados pueden resultar por un lado mucho más espectaculares, y por otro mucho más didácticos o claros, ya que tenemos la capacidad, como humanos, de comprender mucho mejor una animación. Sólo hay que ver la facilidad que tenemos de aprender cosas en una película en comparación con la aptitud que poseemos, en general, de aprender cosas leyendo.

4.4.4. Schemas

Los SCHEMAS (podemos ver con detalle en qué consisten en la Sección 2.1.6) han sido un método de generación de textos muy popular desde su aparición, y lo siguen siendo hoy en día. Sin embargo, suelen presentar estructuras bastante rígidas, y no permites una planificación del texto realmente dinámica.

Nuestro sistema, a pesar de basarse en esta tecnología, añade un poco de dinamismo a estas ideas, añadiendo sobre SCHEMAS la posibilidad de reducirlos, de arriba a abajo, más información que la que se contiene en las reglas de producción del simple SCHEMA. Primero, tenemos que realmente hacemos una búsqueda en profundidad para encontrar el mejor candidato de todos las posibles reducciones, con lo que conseguimos una aplicación más flexible que la que sigue, sin más, un simple patrón de selección de posibles REFERENTES de entrada.

Por otro lado, además, al incluir la percepción de la historia de lector, añadimos capacidades aún mayores de adaptación, ya que en las reglas no simplemente vamos a tener como precondiciones el conocimiento del sistema, sino también todo el contexto anterior de la historia resumido en qué ve el lector de la misma. Por tanto, tenemos que ésta es una de las aportaciones más interesantes de nuestro sistema de generación de historias en lenguaje natural.

4.5. Principales inconvenientes del generador de historias

Desde luego el sistema que hemos desarrollado no es perfecto. Aún tiene ciertos puntos que deben ser observados y mejorados. En esta sección los analizamos.

4.5.1. Tipo de sistema de "arriba hacia abajo"

Nuestro sistema entra dentro del tipo de generadores de lenguaje natural de "arriba hacia abajo", es decir, partiendo de una estructura general, procura ir reduciéndola hasta conseguir un árbol en el que las hojas tienen relación entre sí, con una semántica coherente, y son mensajes concretos. El principal problema de estos sistemas es que no hay manera de garantizar que la información correcta va a formar parte del discurso final, por la misma topología y los algoritmos de la aplicación.

Esto es un problema general que ha de ser resuelto, y probablemente pueda serlo aplicando restricciones en la valoración de historias, haciendo que sea obligatorio incluir ciertos hechos. Sin embargo, es un problema con el que hay que contar y tratar. En la generación de historias no podemos obviar el inconveniente que esto causa, ya que existen ciertos hechos que contienen

información que no está reflejada de ninguna manera en otra parte de la base de conocimiento. Por tanto, no podemos descuidar este aspecto para el trabajo futuro de investigación que seguirá al que presentamos en esta memoria.

4.5.2. Dependencia del dominio

Se ha explicado durante el desarrollo de las ideas principales que los sistemas de GLN tienen el inconveniente principal de ser muy dependientes del sistema cuyos datos van a formar parte de los textos que van a ser generados. Este problema se da principalmente en las partes de la generación relacionadas con la generación del discurso narrativo, y quizás no tanto en la realización superficial y la construcción de la agregación y las expresiones de referencia.

Esta investigación no es una excepción. En la parte de planificación de contenido (el grueso del sistema) tenemos que las posibilidades de la aplicación están muy acotadas respecto de lo que se puede hacer a la hora de contar historias. No en vano, conseguir un sistema que sea capaz de narrar cualquier historia con una calidad suficientemente buena estaría cerca de uno de los objetivos principales de la Ingeniería Lingüística.

Las operaciones de realización superficial que se realizan en nuestro generador de historias, a diferencia de los sistemas más potentes que están plenamente dedicados a a esta parte, sí son dependientes del dominio. Esto se debe principalmente a que hemos adoptado una solución simple y directa, y muy relacionado con lo que queremos recibir de la historia. Por ejemplo, sólo existe una manera de escribir los diálogos, y no es posible hacer construcciones complejas ni referencias a elementos que no existen. Las plantillas que escribimos prácticamente desechan los tiempos verbales. En general, nuestro sistema de realización superficial, si la aplicación fuese dedicada algún día a producción, debería ser reemplazado por otro más potente y general.

Capítulo 5

Conclusiones y trabajo futuro

A lo largo de esta memoria del trabajo de investigación se ha presentado un sistema de generación de historias en lenguaje natural que es capaz, a partir de unos hechos almacenados como una base de conocimiento, crear una historia que describa mediante un texto con características lo más parecidas posible a las humanas la actuación de varios personajes mediante partes habladas y narradas.

Hemos expuesto las tecnologías a partir de las que ha surgido este trabajo, así como las motivaciones que han impulsado que se lleve a cabo. Finalmente hemos comentado, de una manera discutida, las aportaciones principales que ofrece este sistema con relación al trabajo previo sobre el que se apoya.

En este último capítulo sólo nos queda explicar cuál es el estado en el que queda el trabajo y cuáles serán los caminos posteriores de investigación que se seguirán para conseguir que el esfuerzo en la investigación que se ha invertido hasta aquí sea el principio de un trabajo mayor que pueda ofrecer, en la medida de lo posible, un conjunto de nuevas ideas en la Inteligencia Artificial.

Con el prototipo implementado, que sigue las ideas hasta aquí expuestas, pero no dispone de muchas reglas ni tipos de reducción posibles, ahora mismo el generador de historias es capaz de narrar de una manera simple acontecimientos, siempre que se tenga el objetivo del mismo establecido de antemano y se disponga de un conjunto de reglas potente y muy extenso para crear las traducciones. Además, necesitamos una evaluación más completa y estudiada que la que hemos expuesto, ya que sin ella nunca podremos refinar la salida del sistema. Por tanto, aunque es un sistema razonablemente útil, según hemos mostrado con las aplicaciones que hemos contado, y de funcionamiento suficientemente general para dichas aplicaciones, aún queda mucho por hacer.

5.1. Conclusiones sobre el estado actual de la investigación

El sistema que se ha diseñado es el producto de una inversión de estudio y esfuerzo que ha desembocado en la realización de un sistema de generación de historias que ofrece algunas aportaciones a esta disciplina, a partir de las cuales el trabajo puede ser continuado y mejorado. Estas aportaciones han sido comentadas en la Sección 4.1. Sin embargo, éstas no están, ni mucho menos, cerradas o concluidas. Tal y como se ha expuesto en la discusión del sistema, aparacen sobre las caracerísticas originales del sistema ventajas e inconvenientes que no han sido resueltas en este prototipo, y deben seguir siendo estudiadas.

Por tanto, la conclusión general a la que se puede llegar tras el trabajo de investigación es que los resultados han sido, si bien preliminares, sí satisfactorios considerándolos con lo que se esperaba obtener. El objetivo principal de este trabajo era aquél definido en el título: "historias de varios personajes usando narración y diálogo", y esto se ha logrado. En los ejemplos se ha mostrado que la generación de texto que conseguimos no es perfecta, pero sí disponemos de un sistema mejorable que implementa ya ideas importantes sobre las que podemos seguir trabajando.

Por tanto, aunque el trabajo puesto en la realización de la investigación ha dado frutos de interés académico (refrendado por las publicaciones, las cuales exponemos en el Apéndice A), se convierte casi en un deber proseguir con el trabajo realizado hasta aquí, convirtiendo un trabajo de estudios de Máster, mediante la investigación posterior, en un trabajo de Tesis Doctoral. Algunas de las ideas que ya estamos barajando, y sobre las que se ha empezado a trabajar las contamos en las subsiguientes secciones.

5.2. Mejoras del sistema

Desde luego la investigación de este trabajo, a pesar de haber dado frutos en forma de ideas y de pequeñas aportaciones, ha de ser continuada para conseguir objetivos más ambiciosos sobre la generación de historias. El objetivo general de la investigación es conseguir un sistema lo suficientemente potente como para, dentro de un dominio determinado, crear narraciones a partir de varios personajes con una calidad suficiente como para que un humano no tenga la necesidad de llevar a cabo la tarea de escribirlas a partir de los datos. Es mucho el trabajo que queda por hacer, y en esta sección exponemos y proponemos una serie de tareas como trabajo futuro que se tienen planeadas para la continuación de la investigación.

5.3. Mejorar la descripción de los objetivos

Uno de los problemas principales de la generación de textos en Lenguaje Natural es la decisión o establecimiento de qué debe ser narrado. En el comienzo del desarrollo de este trabajo se encontraron dificultades a este respecto. Los primeros prototipos se basaron en la generación de textos a partir de cantidades muy grandes de datos, a partir de los cuales había que generar una historia, filtrando porcentajes muy altos de la entrada ($90\,\% - -95\,\%$). En este punto se afronto la pregunta: *¿qué debemos contar en la historia?*.

En capítulos anteriores ya se ha discutido en más profundidad este tema. Como se ha explicado, tras el estudio de este problema en la literatura correspondiente se ha diseñado una solución, y la propuesta que se ha decidido hacer consiste en establecer unos objetivos cuya semántica operacional está codificada en sus propias reglas de traducción a otros objetivos o a discursos de narración.

A pesar de que esta especificación es suficientemente completa, ya que programamos las reglas en un lenguaje de programación Turing-completo, no es la manera más eficaz de establecerlas, ya que un usuario que quisiera interaccionar con el sistema, con muy poca probabilidad, querría programar cada nuevo objetivo. Por tanto, se concluye que es necesaria otra aproximación a este problema.

Una posible solución, muy directa, sería ofrecer al usuario un abanico de objetivos entre los que él pudiera escoger para recibir una historia o contestación apropiadas. En un sistema más interactiva, hasta podría colegirse este objetivo a partir de otro tipo de entradas provenientes del usuario. No obstante, se prevé como trabajo posterior el encontrar sistemas, ideas y modelos de relación hombre–máquina que permitan crear objetivos de manera dinámica, permitiendo así que el conjunto de historias generado sea mucho mayor, aplicando técnicas de adaptación al sistema de generación de historias.

5.4. Mejorar y aumentar el conjunto de reglas

En cualquier caso, y siguiendo las ideas comentadas en la sección anterior, las reglas seguirían gobernando el comportamiento de nuestro sistema. En este tipo de aplicaciones de la Inteligencia Artificial, el núcleo principal de la investigación está orientado no tanto, desde luego, a la implementación o diseño del programa como a la extracción a partir del *corpus* de las reglas funcionales que definen el sistema, o dicho de otro modo, la algoritmia que suyace (en la mayoría de los casos no presente como tal) detrás del comportamiento humano. Mediante las reglas es posible implementar un modelo del que pudiéramos disponer para establecer la generación de historias, con lo

que a priori resultan suficientemente potentes.

Por tanto, la reducción de objetivos, la creación de los mismos y las maneras de crear el discurso necesitan un conjunto de reglas mucho más potente que éste del que disponemos en el prototipo implementado. Con un conjunto más amplio y elaborado, es seguro que nuestra generación va a ser más rica. El único inconveniente que tiene el uso de reglas es la dificultad de ampliar el sistema, ya que requiere programación por cada funcionalidad nueva de generación que se le quiere aplicar. Este, evidentemente, es un inconveniente de peso que puede provocar un replanteamiento de la aproximación a la resolución del problema distinto al que tenemos ahora.

No obstante, es fundamental apuntar que no existe ninguna seguridad de que los sistemas que sólo intentan emular la generación de textos humana, o cualquier otra disciplina, en general, de la Inteligencia Artificial, puedan ser realmente implementados siguiendo ideas que replican la funcionalidad, y no la topología. Por tanto, a pesar de que como trabajo futuro en esta parte de la investigación se plantea el desarrollo del modelo narrativo y de las reglas correpondientes, no se desestima experimentar y estudiar otras alternativas que creen los textos siguiendo otras técnicas.

5.5. Mejorar el sistema de evaluación

Hemos explicado que el sistema "cierra el lazo" de comunicación con el hombre basándose en un método concreto de evaluación sobre las historias generadas. El método, de forma general, se incluye dentro de aquellos más usados hoy en día, que consisten en la encuesta a un conjunto de personas sobre ciertas características del texto. Estos métodos se aplican así porque no se dispone de un modelo psicológico o siquiera funcional sobre la creación de contenidos de los humanos, por lo que simplemente tenemos que ceñirnos a los comportamientos que podemos comprobar.

Siendo como es nuestra aplicación, la repercusión de estas evaluaciones sobre los textos es muy importante. La búsqueda de la historia correcta que realiza el sistema está totalmente apoyada en estas evaluaciones, de modo que sin ellas, o con un conjunto malo de las mismas, el sistema puede verse seriamente afectado, y, en general, muy degradado.

Como se ha comentado antes, uno de los principales problemas que merece la pena comentar aquí es que, recibida la evaluación, no es evidente cómo aplicarla a la creación de la heurística. Esto se debe a que la "nota" que un encuestado da a un párrafo no depende sólo de ese párrafo, sino de la calidad de los demás, y aislar este dato para analizarlo, no siempre es fácil, dependiendo del caso.

Es por estas razones que como futuro importante de la investigación se procurará, de manera prácticamente obligada, diseñar y poner en práctica un sistema de evaluación más complejo que el que se usa. Muy probablemente

haya que acudir a otros campos de investigación más allá de la tecnología informática o incluso de la lingüística, y sea necesario introducirse en el estudio de la psicología del lenguaje.

5.6. Integración con un sistema de generación de historias completo

Toda esta investigación está englobada en la de desarrollo de un sistema genérico y completo de generación de contenido en lenguaje natural, llamado TAP (Text Arranging Pipeline). Este sistema está dividido en bloques de programación orientados en las fases globales identificadas más comunes de la GLN, a saber: Planificación del contenido, Planificación de las frases y Realización superficial. En la Figura 5.1 queda representada de manera muy simple la arquitectura TAP a grandes rasgos, tal y como sigue la secuencia típica de la GLN. Todo el trabajo desarrollado durante esta investigación ha sido dedicado a la primera de estas tres fases, y la intención general es la de crear un Framework de programas que permita enlazar los trabajos de estos módulos.

Figura 5.1: Diagrama de TAP.

El sistema, por lo tanto, ha sido desarrollado siguiendo las directrices que se han establecido para TAP. En la arquitectura de TAP no sólo se especifican, desde luego, los módulos de alto nivel que han de crearse para el sistema, sino también las interfaces desde el punto de vista estructural que han de cumplirse como comunicación entre los módulos. En estas interfaces se establecen las estructuras de datos que han de ser pasadas, como, por ejemplo, la estructura de árbol que ha de tener un discurso narrativo tras la fase de Planificación de discurso o cómo se representa la salida del Planificador de frases para que la alimentación al realizador superficial sea correcta.

TAP es un sistema de generación de lenguaje natural muy genérico. Se plantea como una arquitectura sobre la que enlazar los módulos que se vayan creando, y estos pueden tener muy diferentes propósitos. Por ejemplo, es posible que el planificador de contenido, módulo que, como se ha comprobado, es muy dependiente de la plataforma, tenga que ser reemplazado según la aplicación. TAP considera este caso, y, a priori, sería posible tener diferentes módulos de planificación conectados a la misma cadena de creación final de texto.

5.7. Creación de una plataforma de comunicación hombre–máquina

Se han presentado en las Secciones 3 y 4.2.1 un sistema de generación de historias en lenguaje natural (el grueso de este trabajo) y una plataforma de barcos robóticos autónomos cooperantes a escala, respectivamente. El desarrollo de ambas investigaciones está estrechamente ligado, y sigue un camino común en el que ambos sistemas puedan complementarse para crear una plataforma en la que el hombre y las máquinas (estos barcos autónomos) puedan comunicarse de una manera en la que el humano perciba los comporamientos y detalles internos de las operaciones en formato de Lenguaje Natural, y no necesite tener experiencia en sistemas informáticos para poder entender "qué está pasando". La Figura 5.2 esquematiza estas ideas.

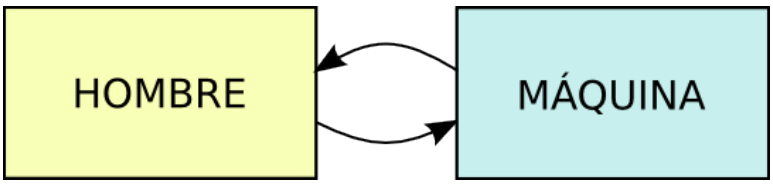

Figura 5.2: Esquema de la relación de comunicación hombre–máquina.

El concepto principal de la investigación sería crear un sistema autónomo de operaciones en el que la plataforma robótica pudiera manteneer informado al responable humano u operario de la maniobra, de modo que su información visual sobre los movimientos de los barcos se viera complementada con textos escritos generados a partir del conocimiento que manejasen los robots. De este modo la aportación principal del sistema sería que se evitaría en gran medida necesitar expertos en los sistemas concretos para entender las operaciones.

La investigación sobre la inclusión de diálogos en los textos narrados está muy motivada, pues, por las características distribuidas de la plataforma robótica. Al tener varios barcos a escala, y basar estos sus capacidades cooperativas en las posibilidades de comunicación y diálogo con las que se les ha dotado, tiene mucho sentido y utilidad ser capaz de transcribir los mensajes "hablados" entre las máquinas como diálogos típicos de los textos literarios a los que estamos acostumbrados, puesto que la aprehensión de estos por parte del hombre es, dada nuestra educación, muy simple y directa. Así, la transmisión desde lo que hacen los robots hacia lo que entienden los humanos, que será el núcleo de la investigación futura en este campo, será más efectiva, entendiendo como tal la relación que existen entre lo que un humano a comprendido de la operación respecto de la operación llevada a cabo por las máquinas. Si esta relación fuese 1, es decir, el humano entiende y conoce toda la operación, esta investigación futura que se propone sería un

éxito. Es más que evidente que este objetivo es muy ambicioso. No obstante, previa puesta en marcha de la investigación en este campo se impone un análisis intenso sobre las posibilidades que en esta materia podemos barajar de cara a realizar, a partir de los conocimientos que hasta aquí hemos expuesto, y los que pudieran ser en un futuro adquiridos, un trabajo de Tesis Doctoral.

Referencias

[App85] D. E. Appelt. Planning english sentences. 1985.

[ASU88] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques and Tools*. Addison-Wesley, 1988.

[BBHC+01] Jill C. Burstein, Lisa Braden-Harder, Martin Chodorow, Bruce A. Kaplan, Chi Lu Karen Kukich, Donald A. Rock, and Susanne Wolff. System and method for computer-based automatic essay scoring. 2001.

[Ber95] E. Bernárdez. *Teoría y Epistemología del Texto*. 1995.

[BF99] S Bringsjord and D Ferrucci. *Artificial Intelligence and Literary Creativity: Inside the mind of Brutus, a StoryTelling Machine*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1999.

[BFM80] L.A. Birnbaum, M. Flowers, and R. McGuire. Towards an ai model of argumentation. In *Proceedings of the 1st AAAI Conference*, pages 195–198, 1980.

[BIR04] M. Belinchón, J. M. Igoa, and A. Revière. *Psicología del lenguaje. Investigación y teoría*. 2004.

[BKW+98] Jill Burstein, Karen Kukich, Susanne Wolfe, Chi Lu, and Martin Chodorow. Enriching automated essay scoring using discourse marking. In *Proceedings of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 15–21, 1998.

[BM03] Jill Burstein and Daniel Marcu. A machine learning approach for identification of thesis and conclusion statements in student essays. pages 455–467, 2003.

[BMF+95] B. Buchanan, J. Moore, D. Forsythe, G. Carenini, G. Banks, and S. Ohlson. An intelligent interactive system for delivering individualized information to patients. *Artificial Intelligence in Medicine*, 7:117–154, 1995.

[Bra87] M. E. Bratman. *Intentions, Plans, and Practical Reason.* Intentions, Plans, and Practical Reason., 1987.

[Cah98] Lynne Cahill. Lexicalisation in applied NLG systems. Technical Report ITRI-99-04, 1998.

[Caw90] A Cawsey. Generating communicative discourse. pages 75–102, 1990.

[CBJ95] A. Cawsey, K Binsted, and R. Jones. Personalised explanations for patient education. In *Proceedings of the 5th European Workshop on Natural Language Generation*, pages 59–74, 1995.

[CIMT00] Dan Cristea, Nancy Ide, Daniel Marcu, and Valentin Tablan. Discourse structure and coreference: An empirical study. In *The 18th International Conference on Computational Linguistics (COLING'00)*, pages 208–214, 2000.

[CL01] Charles B. Callaway and James C. Lester. Narrative prose generation. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1241–1248, Seattle, WA, 2001.

[CL02] Charles B. Callaway and J. C Lester. Narrative prose generation. *Artificial Intelligence*, 139(2):213–252, 2002.

[CMO02] Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. Rst discourse treebank. 2002.

[Dah88] K. Dahlgren. Naive semantics for natural language understanding. 1988.

[Dav78] A. C. Davey. Discourse production. 1978.

[Deh89] Natalie Dehn. Computer story-writing: The role of reconstructive and dynamic memory. Technical Report 792, Departament of Computer Science, Yale University, 1989.

[DHZ95] K. DeSmedt, H. Horacek, and M. Zock. Architectures for natural language generation: Problems and perspectives. In G. Ardoni and M. Zock, editors, *Trends in natural language generation: an artificial intelligence perspective*, LNAI 1036, pages 17–46. Springer Verlag, 1995.

[Dij72] Teun A. van Dijk. Some aspects of text grammars. a study in theoretical linguistics and poetics. 1972.

[dtd07] Dtd tutorial, 2007.

[Elh93] M Elhadad. Fuf: The universal unifier. user manual, version 5.2. Technical Report CUCS-038-91, Columbia University, 1993.

[ER96] M Elhadad and J Robin. An overview of surge: a reusable comprehensive syntactic realization component. Technical Report 96-03, Department of Computer Science, Ben Gurion University, 1996.

[FGGL07] V. Francisco, P. Gervás, M. González, and C. León. Expressive synthesis of read aloud tales. In Patrick Olivier and Christian Kray, editors, *AISB'07, Artificial and Ambient Intelligence, Proceedings of the AISB Annual Convention*, pages 179–186, Culture Lab, Newcastle University, Newcastle upon Tyne, UK, April 2007. The Society for the Study of Artificial Intelligence and Simulation of Behaviour.

[Fox87] Barbara A Fox. Discourse structure and anaphora: Written and conversational english. 1987.

[GDK94] E. Goldberg, N. Driedgar, and R. Kittredge. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9:45–53, 1994.

[Ger01] P. Gervás. An expert system for the composition of formal spanish poetry. pages 181–188, 2001.

[GHST97] Brigitte Grote, Eli Hagen, Adelheit Stein, and Elke Teich. Speech production in human-machine dialogue: A natural language generation perspective. pages 70–85, 1997.

[Gol75] N. Goldman. Conceptual generation. In R. C. Schank, editor, *Conceptual Information Processing*, pages 289–371. North-Holland and Elsevier, Amsterdam and New York, 1975.

[Gri75] J. E. Grimes. The thread of discourse. 1975.

[Gru93] T.R. Gruber. A translation approach to portable ontology specification. pages 199–220, 1993.

[GS86] B. J. Grosz and C. L. Sidner. Attention, intentions, and the structure of discourse. pages 175–204, 1986.

[GSDL06] J.M. Girón-Sierra, A. Dominguez, and C. León. A two linked ships cooperative scenario. In *7th IFAC Conference on Manoeuvring and Control of Marine Craft*, 2006.

[Har01] Elliotte Rusty Harold. *XML Bible*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[HLGH07] Samer Hassan, Carlos León, Pablo Gervás, and Raquel Hervás. A computer model that generates biography-like narratives. In *International Joint Workshop on Computational Creativity. London*, 2007.

[Hob78] J. R. Hobbs. Why is discourse coherent? 1978.

[Hob79] J. R. Hobbs. Coherence and conference. pages 67–90, 1979.

[Hov93] E. Hovy. Automated discourse generation using discourse structure relations. *Artificial Intelligence*, 63(1-2):341–386, 1993.

[HPML07] Samer Hassan, Juan Pavón, Millán Arroyo Menéndez, and Carlos León. Agent based simulation framework for quantitative and qualitative social research: Statistics and natural language generation. In *The Fourth European Social Simulation Association Conference, Tolouse*, 2007.

[IKK+92] L. Iordanska, M. Kim, R. Kittredge, B. Lavoie, and A. Polguère. Generation of extended bilingual statistical reports. In *Proceedings of the Fifteenth International Conference on Computational Linguistics (COLING-92)*, volume 3, pages 1019–1023, 1992.

[KAB+73] Sheldom Klein, J. F Aeschliman, D. F Balsiger, S. L Converse, C Court, M Forster, R Lao, J Oakley, and J Smith. Automatic novel writing: A status report. Technical Report 186, Computer Science Department, University of Wisconsin, 1973.

[Kam81] H. Kamp. A theory of truth and semantic representation. In J. Groenendijk, T. M. V. Janssen, and M. Stokhof, editors, *Truth, Interpretation and Information: Selected Papers from the Third Amsterdam Colloquium*, pages 1–41. Foris Publications, Dordrecht, 1981.

[Kon01] DimitriosÑ Konstantinou. Homer: An intelligent muli-modal story generation system. research plan. `http://www.infm.ulst.ac.uk/~paul/phd/dimitriosprop.doc`, 2001.

[Lev79] D. M. Levy. Communicative goals and strategies: Between discourse and syntax. pages 183–212, 1979.

[LGSE06] C. León, J.M. Girón-Sierra, and S. Esteban. Autonomous scaled ships for experimental study of marine robotics. In *7th IFAC Conference on Manoeuvring and Control of Marine Craft*, 2006.

[LHG07] C. León, S. Hassan, and P. Gervás. From the event log of a social simulation to narrative discourse: Content planning in

story generation. In Patrick Olivier and Christian Kray, editors, *AISB'07, Artificial and Ambient Intelligence, Proceedings of the AISB Annual Convention*, pages 402–409, Culture Lab, Newcastle University, Newcastle upon Tyne, UK, April 2007. The Society for the Study of Artificial Intelligence and Simulation of Behaviour.

[Mar92] J. R. Martin. English text: System and structure. 1992.

[Mar97a] Daniel Marcu. From discourse structures to text summaries. In *Proceedings of ACL Workshop on Intelligent Scalable Text Summarisation*, pages 82–88, 1997.

[Mar97b] Daniel Marcu. From local to global coherence: a bottom-up approach to text planning. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'97)*, 1997.

[Mar97c] Daniel Marcu. The rhetorical parsing, summarization, and generation of natural language texts. 1997.

[Mar00] Daniel Marcu. The theory and practice of discourse parsing and summarization. 2000.

[May90] M. T. Maybury. *Planning Multisentential English Text Using Communicative Acts*. PhD thesis, 1990.

[McD99] David McDonald. Natural Language Generation. 1999.

[McK85] Kathleen R. McKeown. Discourse strategies for generating natural-language text. *Artificial Intelligence*, 27(1):1–41, 1985.

[Mee76] J. Meehan. *The Metanovel: Writing Stories by Computer*. PhD thesis, 1976.

[MFRW00] D.L. McGuinness, R. Fikes, J. Rice, and S Wilder. An environment for merging and testing large ontologies. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference*, 2000.

[MP91] J. D. Moore and C. L. Paris. Discourse structure for explanatory dialogues. In *AAAI Fall Symposium on Discourse*, 1991.

[MS91] J. D. Moore and W. R. Swartout. A reactive account to explanation: taking the users feedback into account. pages 3–48, 1991.

[MT81] W. C. Mann and S. A. Thompson. Computer generation of multiparagraph english text. pages 17–29, 1981.

[MT88] W. Mann and S. Thompson. Rhetorical structure theory: Towards a functional theory of text organization. *Text*, 3:243–281, 1988.

[MT92] William C. Mann and Sandra A. Thompson. *Discourse Description: Diverse linguistic analyses of a fund-raising text.* John Benjamins, Amsterdam, 1992.

[Mus92] M.A Musen. Dimensions of knowledge sharing and reuse. pages 435–467, 1992.

[NM01] Natalya F. Noy and Deborah L. McGuinness. Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Knowledge Systems Laboratory, Stanford University, Stanford, CA, 94305, USA, March 2001.

[OBK06] D. O'Donoghue, A. Bohan, and M. Keane. Seeing things: Inventive reasoning with geometric analogies and topographic maps. 2006.

[O'D00] Michael O'Donnell. Rsttool 2.4: A markup tool for rhetorical structure theory. In *First International Conference on Natural Language Generation (INLG'2000)*, pages 253–256, 2000.

[OM98] Jon Oberlander and Chris Mellish. Final report on the ilex project. 1998.

[OMOK01] Michael O'Donnell, Chris Mellish, Jon Oberlander, and Alistair Knott. Ilex: An architecture for a dynamic hypertext generation system. pages 225–250, 2001.

[OSM94] Kenji Ono, Kazuo Sumita, and Seiji Miike. Abstract generation based on rhetorical structure extraction. In *Proceedings of 15th International Conference on Computational Linguistics (COLING'94)*, pages 344–348, 1994.

[PAHS06] J. Pavón, M. Arroyo, S. Hassan, and C. Sansores. Simulacion de sistemas sociales con agentes software. In *Actas del Campus Multidisciplinar en Percepcion e Inteligencia, CMPI-2006*, volume I, pages 389–400, 2006.

[PG06] Federico Peinado and Pablo Gervás. Evaluation of automatic generation of basic stories. *New Generation Computing, Computational Paradigms and Computational Intelligence. Special issue: Computational Creativity*, 24(3):289–302, 2006.

[Pla] Platón. *Diálogos: Menón.*

[Pro68] Vladimir Propp. *Morphology of the Folk Tale*. 1968.

[PyPS01] Rafael Pérez y Pérez and Mike Sharples. Mexica: a computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence*, 13(2):119–139, 2001.

[PZ03] Livia Polanyi and Annie Zaenen. Shifting attitudes. In *Determination of Information and Tenor in Texts: Multidisciplinary Approaches to Discourse 2003*, 2003.

[RD92] E. Reiter and R. Dale. A fast algorithm for the generation of referring expressions. In *Proceedings of the 14th conference on Computational linguistics*, Nantes, France, 1992.

[RD00] Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge, 2000.

[rdf07] Rdf tutorial, 2007.

[Rei94] E. Reiter. Has a consensus nl generation architecture appeared, and is it psycholinguistically plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 163–170, 1994.

[Rit06] Graeme Ritchie. The transformational creativity hypothesis. 2006.

[RM99] M. Reape and C. Mellish. Just what is aggregation anyway? In *Proceedings of the 7th EWNLG*, Toulouse, France, 1999.

[RML95] E. Reiter, C. Mellish, and J. Levine. Automatic generation of technical documentation. *Applied Artificial Intelligence*, 9:259–287, 1995.

[RS96] Lucia Helena Machado Rino and Donia Scott. A discourse model for gist preservation. 1996.

[Rum72] D. E. Rummelhart. Notes on a schema for stories. 1972.

[Rum75] David E Rumelhart. Notes on a schema for stories. In Daniel G Bobrow and Allan Collins, editors, *Representation and Understanding: Studies in Cognitive Science*, pages 211–236. Academic Press, Inc, New York, 1975.

[RY06] Mark Riedl and R Michael Young. From linear story generation to branching story graphs. *IEEE Journal of Computer Graphics and Applications*, pages 23– 31, 2006.

[Sch69] Roger C Schank. *A conceptual dependency representation for a computer-oriented semantics*. Phd thesis, University of Texas, 1969.

[sem07] Semantic web, 2007.

[She26] H. R. Shepherd. The fine art of writing. 1926.

[Sip97] Michael Sipser. *Introduction to the Theory of Computation*. 1997.

[SJ95] Karen Sparck-Jones. Discourse modelling for automatic summarising. In *Prague Linguistic Circle Papers (Travaux du cercle linguistique de Prague nouvelle série)*, pages 201–227, 1995.

[Swa83] W. Swartout. XPLAIN: a system for creating and explaining expert consulting systems. *Artificial Intelligence*, 21:285–325, 1983.

[Syc87] K. Sycara. Resolving adversarial conflicts: An approach integrating case based and analytical methods. 1987.

[Tab06] M. Taboada. Rhethorical structure theory, 2006.

[Tet05] Joel R Tetreault. Decomposing discourse. In *Anaphora Processing: Linguistic, Cognitive and Computational Modelling*, pages 73–95, 2005.

[TM02] Simone Teufel and Marc Moens. Summarizing scientific articles: Experiments with relevance and rhetorical structure. pages 409–445, 2002.

[TM05] M. Taboada and W. Mann. Applications of rhetorical structure theory. 2005.

[Tur92] Scott R Turner. Minstrel: A computer model of creativity and storytelling. Technical Report UCLA-AI-92-04, Computer Science Department, University of California, 1992.

[Vea06] Tony Veale. Re-representation and creative analogy: A lexico-semantic perspective. 2006.

[Wig06] G. A. Wiggins. Searching for computational creativity. *New Generation Computing, Computational Paradigms and Computational Intelligence. Special issue: Computational Creativity*, 24(3):209–222, 2006.

[wik07] Wikipedia - vladimir propp, 2007.

[Wit22] Ludwig Wittgenstein. *Tractatus Logico-Philosophicus*. 1922.

[YM94] R. Michael Young and Johanna D. Moore. Dpocl: A principled approach to discourse planning. In *Proceedings of the Seventh International Workshop on Natural Langauge Generation, Kennebunkport*, 1994.

Apéndice A

Publicaciones

El desarrollo de este proyecto ha dado lugar a las siguientes publicaciones. Están expuestas aquellas relacionadas directamente con esta investigación, y con la experimentación de barcos cooperantes a escala, que se ha realizado en paralelo con esta, y ha servido como fuente de datos para las historias:

- **From the Event Log of a Social Simulation to Narrative Discourse: Content Planning in Story Generation**. León, C.; Hassan, S. & Gervas, P. AISB'07, Artificial and Ambient Intelligence, Proceedings of the AISB Annual Convention, 2007, 402-409.

  Este artículo presenta la primera versión de Heródoto (Sección 4.2.2). Se muestra cómo el sistema de simulación social correspondiente se adaptó al dominio apropiado. También se presentan las principales ideas que conforman la primera versión del Determinador de Contenido y del Planificador de discurso. Asimismo se muestra un Realizador superficial simple, que sirvió de base para versiones más desarrolladas del mismo. Se estudia cómo un sistema de Lenguaje Natural de este tipo puede ayudar en sistemas de gran generación de datos de muchos personajes, como por ejemplo juegos masivos multijugador en Internet.

- **A Computer Model that Generates Biography-Like Narratives**. Samer Hassan, Carlos León, Pablo Gervás, Raquel Hervás. International Joint Workshop on Computational Creativity, 2007.

  Centrado más en la creatividad que se podía percibir del sistema, este artículo presentó las capacidades de emulación de la creatividad del humano por parte del sistema de generación. Se estableción un estudio sobre la incidencia, además, de los parámetros de la simulación en la generación de los datos del mundo. Una parte muy importante de este trabajo es la discusión sobre la calidad de la creatividad que puede alcanzarse con un sistema de generación como el que se expuso. Se

muestra una segunda versión de HERÓDOTO en la que se hace más énfasis en la PLANIFICACIÓN DEL DISCURSO, presentado la idea del *foco* sobre los personajes para añadir la capacidad de contar historias de más de un personaje.

- **Agent Based Simulation Framework for Quantitative and Qualitative Social Research: Statistics and Natural Language Generation**. Samer Hassan, Juan Pavón, Millán Arroyo Menéndez y Carlos León. The Fourth European Social Simulation Association Conference, Tolouse, 2007.

  Orientado a la pura simulación social, en este trabajo se presenta cómo, aparte de un enfoque de estudio macrosocial mediante simplifación para el análisis usando técnicas estadísticas, es posible usar un resumen en lenguaje natural sobre la vida de uno de los individuos de la simulación para realizar un análisis cualitativo (enfoque *micro*).

- **Robotics and Automation in the Maritime Industries, Chapter 14: Towards automatized cooperation of ships in spill-over recovery scenarios**. J. M. de la Cruz, J. Giron-Sierra, S. Esteban, J. Recas, J. F. Jiménez, S. Cifuentes, C. Leon, E. Besada, J. A. Lopez-Orozco, B. de Andres and J. Fdez. Prisuelos. Editado por J. Aranda, P. González de Santos y J. M. de la Cruz. Páginas 305-335, ISBN-13: 978-84-611-3915-6, ISBN-10: 84-611-3915-1.

  En este capítulo de libro se muestra un resumen del trabajo del grupo de investigación de Ingenería de Sistema y Automática con los barcos cooperativos autónomos a escala, dando detalle de su construcción y funcionamiento, así como el estudio matemático y físico de sus capacidades de maniobrabilidad.

- **Autonomous scaled ships for experimental study of marine robotics**. León, C.; Girón-Sierra, J. & Esteban, S. 7th IFAC Conference on Manoeuvring and Control of Marine Craft, 2006.

  El trabajo que se muestra en esta publicación muestra de manera detallada cuáles son las partes lógicas y físicas de los barcos a escala, mostrando sus capacidades cooperativas preliminares usando un sistema de lenguaje entre máquinas diseñado al efecto, conocido como *Ship Language*.

- **A two linked ships cooperative scenario**. Giron-Sierra, J.; Dominguez, A. & Leon, C. 7th IFAC Conference on Manoeuvring and Control of Marine Craft, 2006.

  En este artículo se muestra un estudio sobre la captura de una mancha de petróleo en el mar usando un grupo de barcos que han de trabajar de manera conjunta, unidos por una red de arrastre. Se hace énfasis

en las leyes de control que han de seguir las naves para que la operación sea exitosa, y se muestra un experimento realizado con los barcos autónomos a escala.

Apéndice B

Ejemplo de XML de entrada

En este apéndice incluimos un ejemplo de un XML de entrada correspondiente a una versión del prototipo anterior a la que se presenta como programa final y que se explica en la Sección 4.2.2. La intención de este apéndice es mostrar cómo de largo e ilegible puede ser un conjunto de datos de entrada, y, de este modo, justificar la utilidad de un sistema de ordenación y filtro que es nuestro generador de historias en lenguaje natural.

A continuación tenemos un sólo un fragmento del archivo XML, ya que todo el archivo de entrada sería demasiado largo. Después del código XML se ha incluido una generación posible realizada con un prototipo de la aplicación anterior al presentado, pero que sigue las mismas ideas. En este conjunto de datos se incluye también la información sobre el modelo BDI que tienen los personajes, lo que ayudó, en este prototipo, para crear relaciones de causalidad entre los diferentes hechos aplicando un conjunto simple de reglas.

```
<Story Id="fantasy">
 <Description>
 A fantasy Middlee-Age world
 </Description>
 <Log Id="i8">
    <Attribute Id="name" Value="derona"/>
    <Attribute Id="last_name" Value="cairnbreaker"/>
    <Attribute Id="race" Value="dwarf"/>
    <Attribute Id="sex" Value="male"/>
    <Attribute Id="age_end" Value="42"/>
    <Attribute Id="age_state_end" Value="adult"/>
    <Attribute Id="ideology" Value="left"/>
    <Attribute Id="education" Value="high"/>
    <Desire Type="kill" Object="dragon"/>
    <Desire Type="become" Object="wizard"/>
    <Desire Type="find" Object="lost brother"/>
    <Event Id="e1" Time="499" Action="birth" Param=""/>
    <Event Id="e2" Time="501" Action="enemy" Param="i1"/>
```

```
   <Event Id="e3" Time="504" Action="enemy" Param="i3"/>
   <Event Id="e4" Time="506" Action="saved" Param="priest"/>
   <Event Id="e5" Time="506" Action="increased"
          Param="religion"/>
   <Event Id="e6" Time="513" Action="friend" Param="i10"/>
   <Event Id="e7" Time="515" Action="grow" Param="adult"/>
   <Event Id="e8" Time="515" Action="SAY"
          Param="B:who has:the one ring:i10"/>
   <Event Id="e9" Time="515" Action="friend" Param="i5"/>
   <Event Id="e10" Time="518" Action="friend" Param="i6"/>
   <Event Id="e11" Time="518" Action="SAY"
          Param="B:who has:the one ring:i6"/>
   <Event Id="e12" Time="519" Action="friend" Param="i9"/>
   <Event Id="e13" Time="520" Action="enemy" Param="i2"/>
   <Event Id="e3" Time="503" Action="spelled" Param="memory"/>
   <Event Id="e4" Time="503" Action="decreased"
          Param="education"/>
   <Event Id="e14" Time="524" Action="involved" Param="battle"/>
   <Event Id="e15" Time="527" Action="enemy" Param="i12"/>
   <Event Id="e16" Time="527" Action="spelled" Param="fireball"/>
   <Event Id="e17" Time="529" Action="kill" Param="ogre"/>
   <Event Id="e18" Time="532" Action="KNOW"
          Param="B:where another:wizard:i6"/>
   <Event Id="e19" Time="533" Action="SAY"
          Param="B:where another:wizard:i9"/>
   <Event Id="e20" Time="534" Action="SAY"
          Param="B:who has:the one ring:i5"/>
   <Event Id="e21" Time="541" Action="mysterious accident"
          Param=""/>
</Log>

[...]

<Log Id="i5">
   <Attribute Id="name" Value="parbagar"/>
   <Attribute Id="last_name" Value="greatcutter"/>
   <Attribute Id="race" Value="orc"/>
   <Attribute Id="sex" Value="male"/>
   <Attribute Id="age_end" Value="50"/>
   <Attribute Id="age_state_end" Value="adult"/>
   <Attribute Id="ideology" Value="left"/>
   <Attribute Id="education" Value="very low"/>
   <Desire Type="kill" Object="dragon"/>
   <Desire Type="find" Object="the one ring"/>
   <Desire Type="become" Object="wizard"/>
   <Event Id="e1" Time="499" Action="birth" Param=""/>
   <Event Id="e2" Time="500" Action="I ALREADY KNOW"
          Param="B:where:the one ring"/>
   <Event Id="e3" Time="503" Action="spelled" Param="memory"/>
```

```
   <Event Id="e4" Time="503" Action="decreased" Param="education"/>
   <Event Id="e5" Time="505" Action="involved" Param="battle"/>
   <Event Id="e6" Time="506" Action="enemy" Param="i9"/>
   <Event Id="e7" Time="506" Action="enemy" Param="i7"/>
   <Event Id="e8" Time="512" Action="inherit" Param="castle"/>
   <Event Id="e9" Time="512" Action="increased" Param="economy"/>
   <Event Id="e10" Time="512" Action="lost" Param="labyrinth"/>
   <Event Id="e11" Time="515" Action="grow" Param="adult"/>
   <Event Id="e12" Time="515" Action="friend" Param="i8"/>
   <Event Id="e13" Time="516" Action="inherit" Param="castle"/>
   <Event Id="e14" Time="516" Action="increased" Param="economy"/>
   <Event Id="e15" Time="516" Action="involved" Param="battle"/>
   <Event Id="e16" Time="524" Action="impossible love" Param=""/>
   <Event Id="e17" Time="526" Action="friend" Param="i3"/>
   <Event Id="e18" Time="526" Action="spelled" Param="memory"/>
   <Event Id="e19" Time="526" Action="decreased" Param="education"/>
   <Event Id="e20" Time="528" Action="lost" Param="labyrinth"/>
   <Event Id="e21" Time="530" Action="involved" Param="battle"/>
   <Event Id="e22" Time="534" Action="spelled" Param="fireball"/>
   <Event Id="e23" Time="534" Action="KNOW"
          Param="B:who has:the one ring:i8"/>
   <Event Id="e24" Time="538" Action="DO"
          Param="I:travel:wild lands"/>
   <Event Id="e25" Time="539" Action="DO"
          Param="I:find:gollum's cave"/>
   <Event Id="e26" Time="539" Action="dungeon"
          Param="the one ring"/>
   <Event Id="e27" Time="540" Action="DO"
          Param="I:trick:gollum"/>
   <Event Id="e28" Time="543" Action="DO"
          Param="I:find:the one ring"/>
   <Event Id="e29" Time="543" Action="ACCOMPLISHED"
          Param="D: find:the one ring"/>
   <Event Id="e30" Time="544" Action="KNOW"
          Param="B:where another:wizard:i3"/>
   <Event Id="e31" Time="547" Action="friend" Param="i2"/>
   <Event Id="e32" Time="549" Action="betrayed" Param="i2"/>
</Log>

[...]

<Log Id="i9">
   <Attribute Id="name" Value="georgia"/>
   <Attribute Id="last_name" Value="houston"/>
   <Attribute Id="race" Value="human"/>
   <Attribute Id="sex" Value="female"/>
   <Attribute Id="age_end" Value="51"/>
   <Attribute Id="age_state_end" Value="adult"/>
   <Attribute Id="ideology" Value="very right"/>
```

```
    <Attribute Id="education" Value="very high"/>
    <Desire Type="become" Object="wizard"/>
    <Desire Type="find" Object="lost brother"/>
    <Desire Type="kill" Object="dragon"/>
    <Event Id="e1" Time="499" Action="birth" Param=""/>
    <Event Id="e2" Time="500" Action="I ALREADY KNOW"
           Param="B:weak point:dragon"/>
    <Event Id="e3" Time="500" Action="friend" Param="i4"/>
    <Event Id="e4" Time="501" Action="saved" Param="priest"/>
    <Event Id="e5" Time="501" Action="increased"
           Param="religion"/>
    <Event Id="e6" Time="502" Action="friend" Param="i10"/>
    <Event Id="e7" Time="505" Action="dungeon" Param="potion"/>
    <Event Id="e8" Time="505" Action="increased"
           Param="strength"/>
    <Event Id="e9" Time="506" Action="enemy" Param="i5"/>
    <Event Id="e10" Time="515" Action="grow" Param="adult"/>
    <Event Id="e11" Time="515" Action="friend" Param="i7"/>
    <Event Id="e12" Time="515" Action="friend" Param="i6"/>
    <Event Id="e13" Time="517" Action="involved"
           Param="battle"/>
    <Event Id="e14" Time="518" Action="friend" Param="i3"/>
    <Event Id="e15" Time="519" Action="friend" Param="i8"/>
    <Event Id="e16" Time="520" Action="SAY"
           Param="B:weak point:dragon:i6"/>
    <Event Id="e17" Time="522" Action="couple" Param="i7"/>
    <Event Id="e18" Time="522" Action="child" Param="i11"/>
    <Event Id="e19" Time="522" Action="child" Param="i12"/>
    <Event Id="e20" Time="522" Action="dungeon"
           Param="treasure"/>
    <Event Id="e21" Time="522" Action="increased"
           Param="economy"/>
    <Event Id="e22" Time="533" Action="KNOW"
           Param="B:where another:wizard:i8"/>
    <Event Id="e23" Time="540" Action="kill" Param="ogre"/>
    <Event Id="e24" Time="543" Action="SAY"
           Param="B:where another:wizard:i3"/>
    <Event Id="e25" Time="544" Action="SAY"
           Param="B:where another:wizard:i4"/>
    <Event Id="e26" Time="546" Action="impossible love"
           Param="i4"/>
 </Log>

 [...]

</Story>
```

A continuación, el texto generado correspondiente. Ha sido generado eligiendo dos personajes que han sido considerados principales, y ajustando

las reglas para que se cree el diálogo que satisface algunos de sus deseos como personajes.

> *It was a man. And His name was Deron. And His last name was Cairnbreaker. And Deron Cairnbreaker desired to become a great wizard. After that, the spell of memory was cast upon Deron Cairnbreaker. Because of that, its education decreased. After that, Deron Cairnbreaker and Parbagar Greatcutter talked:*
>
> *– Do you know who has the one ring?*
>
> *– Yes, I can tell you who has the one ring - said Deron Cairnbreaker, and it told where.*
>
> *– Are you sure? Then I'll go and talk with it. - said Parbagar Greatcutter - Farewell.*
>
> *Before that, Deron Cairnbreaker and Georgia Houston talked:*
>
> *– Do you know where can I find another wizard?*
>
> *– Yes, I do. I will tell you. - said Deron Cairnbreaker.*
>
> *Then, Deron Cairnbreaker showed the place.*
>
> *– Ok, now I have this useful information. - said Georgia Houston -Thank you!*

Apéndice C

Entrada para el fragmento del Menón

Este código en el lenguaje creado para el prototipo presentado de la aplicación contiene la información sobre el fragmento del diálogo de Menón escrito por Platón, que se ha creado a mano para la demostración del sistema de generación de historias en lenguaje natural. El texto generado puede verse en la Sección 3.4.

```
#
# Elementos de la historia del diálogo
# de Menón (fragmento)
#

charater socrates
character meno
character people
character gorgias

entity virtue
entity old
entity idea-of-virtue
entity philosopher
entity multiple
entity bees
entity unique
entity courage
entity temperance
entity wisdom
entity magnanimity
entity confused

entity yes
entity no
```

```
location thessaly
location athens

verb ask
verb say
verb ask_for
verb negate
verb acquire
verb learn
verb practice
verb agree
verb think
verb tell
verb incoherent
verb located
verb know

property p1 {
    who socrates;
    verb be;
    attribute old;
    init start;
    end theend
}

property p2 {
    who socrates;
    verb likes;
    attribute meno;
    init start;
    end theend
}

property p3 {
    who meno;
    verb likes;
    attribute socrates;
    init start;
    end theend;
}

property p4 {
    who socrates;
    verb be;
    attribute philosopher;
    init start;
    end theend
}
```

```
event preev0 {
    who meno, socrates;
    verb located;
    location athens;
    init start;
    end theend;
}

event evp2 {
    who meno;
    verb think;
    direct event evp1 {
        who socrates;
        verb know;
        direct virtue;
        init start;
        end theend;
    }
}

property pror {
    who ev2;
    verb incoherent;
    attribute ev3;
}

dialogue d1 {
    who meno;
    verb ask_for;
    indirect socrates;
    direct event ev1 {
        who socrates;
        verb tell;
        indirect meno;

        direct event ev2 {
            who people;
            verb acquire;
            direct virtue;
            because event ev4 {
                who people;
                verb learn;
                direct virtue;
            };
        },
        event ev3 {
            who people;
            verb acquire;
            direct virtue;
```

```
            because event ev5 {
                who people;
                verb practice;
                direct virtue;
            };
        };
    };
}

dialogue d2 {
    who socrates;
    verb negate;
    indirect meno;
    direct event ev6 {
        who socrates;
        verb know;
        direct virtue;
    };
}

dialogue d3 {
    who meno;
    verb ask;
    indirect socrates;
    direct event ev1d3 {
        who meno;
        verb must;
        direct event ev7  {
            who meno;
            verb tell;
            location thessaly;
            direct d2;
        };
    }
}

dialogue d4 {
    who socrates;
    indirect meno;
    verb say;
    direct yes;
}

dialogue d5 {
    who meno;
    indirect socrates;
    verb ask;
    direct event ev8 {
        who socrates;
```

```
        verb meet;
        direct gorgias;
        init before_now;
        end before_now;
        location athens;
    };
}

dialogue d6 {
    who socrates;
    indirect meno;
    verb say;
    direct yes;
}

dialogue d7 {
    who meno;
    indirect socrates;
    verb say;
    direct event ev9 {
        who meno;
        verb think;
        direct idea-of-virtue;
        because event ev10 {
            who gorgias;
            verb think;
            direct idea-of-virtue;
        };
    };
}

dialogue d8 {
    who socrates;
    verb say;
    indirect meno;
    direct event ev11 {
        who meno;
        verb tell;
        direct idea-of-virtue;
        indirect socrates;
    };
}

dialogue d9 {
    who meno;
    indirect socrates;
    verb say;
    direct property pr1d9 {
        who virtue;
```

```
        verb be;
        attribute multiple;
    };
}

dialogue d10 {
    who socrates;
    verb say;
    indirect meno;
    direct property pr1d10 {
        who d9;
        verb incoherent;
        attribute property pr2d10 {
            who bees;
            verb be;
            verb unique;
        };
    };
}

property pr7 {
    who meno;
    verb be;
    attribute confused;
}

dialogue d11 {
    who meno;
    indirect socrates;
    direct property {
        who courage, temperance, wisdom, magnanimity;
        verb be;
        attribute virtue;
    }
}

dialogue d12 {
    who socrates;
    indirect meno;
    direct event ev12 {
        who god;
        verb give;
        direct virtue;
        indirect people;
    }
}

dialogue d13 {
    who meno;
```

```
    indirect socrates;
    direct event ev13 {
        who meno;
        verb agree;
        direct socrates;
    }
}
```

Apéndice D

Entrada para el ejemplo de los barcos autónomos

Este código corresponde a un diálogo entre dos barcos que arrastran una red y tienen que realizar un giro complicado de manera cooperativa, y corresponden al texto de la Sección 4.2.1.

```
#
# Elementos de la historia de dos barcos
# cooperativos
#

character ship-1
character ship-2

entity net
entity message
entity method
entity orden
entity dificil
entity giro
entity following
entity ok
entity direction

location location-a
location location-b
location location-c
location location-d

verb be
verb ask
verb know
verb move
```

```
verb type
verb beattached
verb receive
verb send
verb control
verb say
verb stop

property pr1 {who message; verb be;
  attribute orden; init start; end theend}

property pr2 {who message; verb type;
  attribute giro; init start; end theend}

property pr3 {who ship-1; verb beattached;
  attribute net; init start; end theend}

property pr4 {who ship-2; verb beattached;
   attribute net; init start; end theend}

event ev1 {who ship-1; verb located;
  where location-a; init 1; end 2}

event ev2 {who ship-2; verb located;
  where location-b; init 1; end 2}

event ev3 {who ship-1; verb receive;
  init 2; end 3; direct message}

event ev5 {who ship-1; verb turn; direct
  location-c; init 3; end 4;}

dialog di1 {who ship-1; verb say; indirect
   ship-2; direct ev3; init 2; end 3}

dialog di2 {who ship-1; verb say; indirect
  ship-2; direct ev5; init 3; end 4; because
  ev3, pr3, pr4}

property pr5 {who method; verb be;
  attribute following; init start; end theend}

event ev6 {who ship-1; verb know;
  direct method; init 3; end 4;}

dialog di3 {who ship-2; verb ask; indirect
  ship-1; direct ev6; init 4; end 5}

dialog di4{who ship-1; verb say; indirect
```

```
  ship-2; direct pr5;  init 5; end 6; because ev6}

dialog di5{who ship-2; verb say; indirect
  ship-1; direct ok;  init 6; end 7}

event ev7{who ship-1; verb move; where
  location-d; init 7; end 8; because ev3}

event ev8{who ship-2; verb move; where
  location-c; init 7; end 8; because ev3}

event ev9{who ship-1; verb control;
  direct direction; indirect ship-1; init 8; end 9}

event ev10{who ship-1; verb stop; init
   9; end 10; because ev3}

event ev11{who ship-2; verb stop; init
  9; end 10; because ev3}
```